M-Class
Waveform Generator

Models ZT5211, ZT5212

User's Manual: 0004-000074
Revision 1a

November 30, 2009

Contact

ZTEC Instruments
7715 Tiburon Street NE
Albuquerque, NM 87109

Telephone: (505) 342-0132
Fax: (505) 342-0222
Web Site: www.ztecinstruments.com

ZTEC Instruments, Inc. welcomes your comments on this manual. All manuals are thoroughly reviewed before distribution. We are, however, grateful for any comments from our users which will further help to improve the content and quality of our documents.

Copyright

Copyright 2009 by ZTEC Instruments

Printed in the United States of America.

All rights reserved under copyright laws of the United States and other countries.

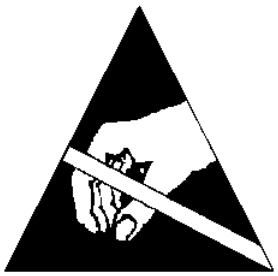
All technical data and computer software documentation contained herein is proprietary and confidential to ZTEC Instruments, Inc. or its licensor. The reproduction and/or transmission of this publication in whole or in part by any means, electronic or mechanical, is prohibited without the prior written consent of ZTEC Instruments, Inc.

ZTEC and the ZTEC logo are registered trademarks of ZTEC Instruments.

ZTEC Instruments has attempted throughout this publication to distinguish proprietary trademarks from descriptive terms by following the capitalization style used by the manufacturer. Product names listed are trademarks of their respective manufacturers. Company names listed are trademarks or trade names of their respective companies.

LabVIEW™	National Instruments Corporation
Windows™	Microsoft Corporation

The material in this manual is for informational purposes only and is subject to change without notice. ZTEC Instruments, Inc. assumes no responsibility for any error or for consequential damages that may result from the use or misinterpretation of any of the procedures in this publication.



Handling Precautions for Electronic Devices Subject to Damage by Static Electricity

This instrument is susceptible to Electronic Static Discharge (ESD) damage. When transporting, place the instrument or module in conductive (anti-static) envelopes or carriers. Open only at an ESD-approved work surface. An ESD safe work surface is defined as follows:

- The work surface must be conductive and reliably connected to an earth ground with a safety resistance of approximately 250 kilo Ohms.

- The surface must NOT be metal. A resistance of 30–300 kilo Ohms per square inch is suggested.

Ground the frame of any line-powered equipment, chassis, test instruments, lamps, soldering irons, etc., directly to the earth ground. To avoid shorting out the safety resistance, ensure that the grounded equipment has rubber feet or other means of insulation from the work surface.

Avoid placing tools or electrical parts on insulators. Do NOT use any hand tool that can generate a static charge, such as a non-conductive plunger-type solder sucker. Use a conductive strap or cable with a wrist cuff to reliably ground to the work surface. The cuff must make electrical contact directly with the skin; do NOT wear it over clothing.

Note: Resistance between the skin and the work surface is typically 250 kilo Ohms to 1 mega Ohm using a commercially-available personnel grounding device.

Avoid circumstances that are likely to produce static charges, such as wearing clothes of synthetic material, sitting on a plastic-covered stool (especially when wearing woolen material), combing the hair, or making extensive pencil erasures. These circumstances are most significant when the air is dry.

When testing static sensitive devices, ensure DC power is ON before, during, and after application of test signals. Ensure all pertinent voltages are switched OFF while circuit boards or components are removed or inserted.

Revision History

Rev	Date	Section	Description
1	6-9-2009	All	Initial Release
1a	11-30-2009	LXI, terminology	Added socket section, added terminology section

Table of Contents

Introduction 15
- Description 15
- Product Options and Platforms 15
- Front Panel 17
- Back Panel 20
- Terminology 21
- Additional Resources 21

Functionality and Operation 22
- Functional Block Diagram 22
- Data Flow 23
- Clock Generation and Synchronization 24
  - DAC Sampling Clock 24
  - Time Base Reference Clock 24
- Waveform Types 25
  - Standard Functions 25
  - Arbitrary Waveforms 25
  - Waveform Sequences 26
- Waveform Memory 27
  - Waveform Size and Cycle Period 27
    - Standard Function Size 28
    - Arbitrary Waveform Size 29
  - Dual DAC Memory Banks 29
  - Waveform Switching 30
  - Arbitrary Waveform Library 30
  - Reference Waveforms 31
  - Waveform Handles 31
  - Waveform Data Format 31
  - Waveform Operations 31
    - Waveform Upload 31
    - Waveform Download 32
    - Waveform Copy 32
    - Waveform Invert 32
    - Waveform Scale 32
- Operation Modes 33
  - Continuous Mode 33
  - Burst Mode 33
  - Binary Modulation Mode 34
  - Sweep Mode 34
- Standard Function Descriptions 35
  - Sine 35
  - Square 35
  - Triangle 36
  - Ramp Up 36
  - Ramp Down 37
  - DC 37
  - Haversine 37
  - Havercosine 38
  - Half Cycle Sine 38

Pulse 39
Sinc Pulse 39
Gaussian Pulse 40
Lorentz Pulse 40
Periodic Random Noise 41
AM 41
FM 41
Multi-Tone 42
Serial Data 43
External Input 44
Trigger and Arm 44
Trigger Initiate Model 44
Trigger Conditions 45
Trigger Processing 46
Pattern 46
Internal Trigger 47
Software Trigger 47
Trigger Timestamp 47
Arm 47
Output Channels 48
Output Channel Enable 48
Output Signal Conditioning 48
Output Impedance 48
Output Amplitude 49
Output DC Offset 49
Output Filter 49
Sync & Trigger Outputs 50
Sync Pulse Generation 50
External Output 50
Event Outputs 50
Utilities and Status Reporting 51
Reset 51
Undo 51
Save and Recall States 51
Error 51
Status 51
Condition 53
Event 53
Enable 53
Self Test 54
Calibrate 54
Instrument Calibration 54
Identification and Version 55

Interface 56
PCI/PXI Interface 56
PXI Interoperability for the ZT5211PXI 56
PCI Interoperability for the ZT5211PCI 56
PCI Timing Expansion Connector 56
VXI Interface 57
Interface Description 57

LXI Interface 58
Interface Description 58
webLXI 58
IP Address 58
Ports 59
Sockets 59
NTP Server Address 59
Hardware Triggering 59
Driven Mode 60
Wired-Or Mode 60
Connections 60
Software 61
Command Reference 62
Common Command Format 62
SCPI Command Format 62
Command Separator 63
Abbreviated Commands 63
Implied Commands 63
Parameters 64
Optional Parameters 65
Parameters Out of Range 65
Linking Commands 65
Linking IEEE 488.2 Common Commands with SCPI Commands 65
Linking Multiple SCPI Commands 65
IEEE 488.2 Common Commands 66
Clear Status Command 66
Event Status Enable Command 66
Event Status Enable Query 66
Event Status Register Query 67
Identification Query 67
Operation Complete Command 68
Operation Complete Query 68
Recall Instrument State Command 68
Reset Command 68
Save Instrument State Command 69
Service Request Enable Command 69
Service Request Enable Query 69
Status Byte Query 70
Trigger Immediate Command 70
Test Query 71
Wait to Continue Command 71
SCPI Commands and Queries 72
Abort Command 72
Accessory Identify Query 72
AM Depth Command 73
AM Depth Query 73
AM Frequency Command 73
AM Frequency Query 73
AM Shape Command 74
AM Shape Query 74

Arm Command 74
Arm Query 74
Arm Polarity Command 75
Arm Polarity Query 75
Arm Source Command 75
Arm Source Query 75
Binary Modulation View Command 76
Binary Modulation View Query 76
Binary Modulation Source Command 76
Binary Modulation Source Query 76
Binary Modulation State Command 77
Binary Modulation State Query 77
Burst Count Command 77
Burst Count Query 77
Calibration Date Query 78
Calibration Default Command 78
Calibration Restore Command 78
DAC Clock Common Command 79
DAC Clock Common Query 79
DAC Clock Frequency Command 79
DAC Clock Frequency Query 79
DAC Clock Mode Command 80
DAC Clock Mode Query 80
Duty Cycle Command 80
Duty Cycle Query 80
Filter Frequency Command 81
Filter Frequency Query 81
FM Deviation Command 82
FM Deviation Query 82
FM Frequency Command 83
FM Frequency Query 83
FM Shape Command 84
FM Shape Query 84
Format Byte Order Command 84
Format Byte Order Query 84
Format Data Command 85
Format Data Query 85
Format Precision Command 85
Format Precision Query 85
Frequency Command 86
Frequency Query 86
Function Shape Command 87
Function Shape Query 87
Gaussian Standard Deviation Command 88
Gaussian Standard Deviation Query 88
Initiate Continuous Command 88
Initiate Continuous Query 88
Initiate Command 89
Initiate Query 89
Lorentz Half Width Command 89
Lorentz Half Width Query 89

Multi-Tone Default Command 90
Multi-Tone Tone Frequency Command 90
Multi-Tone Tone Frequency Query 90
Multi-Tone Tone State Command 91
Multi-Tone Tone State Query 91
Operation Mode Command 92
Operation Mode Query 92
Output ECL Trigger Polarity Command 93
Output ECL Trigger Polarity Query 93
Output ECL Trigger Source Command 94
Output ECL Trigger Source Query 94
Output ECL Trigger State Command 95
Output ECL Trigger State Query 95
Output Event Time Command 95
Output Event Time Query 95
Output External Polarity Command 96
Output External Polarity Query 96
Output External Pulse Mode Command 96
Output External Pulse Mode Query 96
Output External Pulse Period Command 97
Output External Pulse Period Query 97
Output External Source Command 97
Output External Source Query 97
Output External State Command 98
Output External State Query 98
Output LXI Mode Command 98
Output LXI Mode Query 98
Output LXI Polarity Command 99
Output LXI Polarity Query 99
Output LXI Source Command 100
Output LXI Source Query 100
Output LXI State Command 101
Output LXI State Query 101
Output Mode Command 101
Output Mode Query 101
Output Source Query 102
Output State Command 102
Output State Query 102
Output Sync Off Position Command 103
Output Sync Off Position Query 103
Output Sync On Position Command 103
Output Sync On Position Query 103
Output TTL Trigger Polarity Command 104
Output TTL Trigger Polarity Query 104
Output TTL Trigger Source Command 105
Output TTL Trigger Source Query 105
Output TTL Trigger State Command 106
Output TTL Trigger State Query 106
Period Command 107
Period Query 107
Phase Command 108

Phase Query 108
Pulse Transition Leading Command 108
Pulse Transition Leading Query 108
Pulse Transition Trailing Command 109
Pulse Transition Trailing Query 109
Reference Oscillator Frequency Query 109
Reference Oscillator Output State Command 110
Reference Oscillator Output State Query 110
Reference Oscillator Source Command 110
Reference Oscillator Source Query 110
Sense ECL Trigger State Query 110
Sense LXI State Query 111
Sense TTL Trigger State Query 111
Sequence Clear Command 111
Sequence Clear All Command 112
Sequence Data Loop Count Command 112
Sequence Data Loop Count Query 112
Sequence Data Waveform Command 112
Sequence Data Waveform Query 112
Sequence Generate Query 113
Sequence Loop Maximum Query 113
Sequence Maximum Query 113
Sequence Size Command 114
Sequence Size Query 114
Sequence Size Maximum Query 114
Sequence Size Minimum Query 114
Serial Data Bit Period Command 115
Serial Data Bit Period Query 115
Serial Data Word Length Command 115
Serial Data Word Length Query 115
Serial Data Word Command 116
Serial Data Word Query 116
Sinc Frequency Command 116
Sinc Frequency Query 116
Status Interrupt Request State Command 117
Status Interrupt Request State Query 117
Status Operation Condition Query 118
Status Operation Enable Command 119
Status Operation Enable Query 119
Status Operation Event Query 120
Status Preset Command 121
Status Questionable Calibration Condition Query 121
Status Questionable Calibration Enable Command 122
Status Questionable Calibration Enable Query 122
Status Questionable Calibration Event Query 123
Status Questionable Condition Query 124
Status Questionable Enable Command 125
Status Questionable Enable Query 125
Status Questionable Event Query 126
Status Questionable Frequency Condition Query 127
Status Questionable Frequency Enable Command 128

Status Questionable Frequency Enable Query 128
Status Questionable Frequency Event Query 129
Status Questionable Test Condition Query 130
Status Questionable Test Enable Command 131
Status Questionable Test Enable Query 131
Status Questionable Test Event Query 132
Status Questionable Test AWG Condition Query 133
Status Questionable Test AWG Enable Command 134
Status Questionable Test AWG Enable Query 134
Status Questionable Test AWG Event Query 135
Status Questionable Voltage Condition Query 136
Status Questionable Voltage Enable Command 137
Status Questionable Voltage Enable Query 137
Status Questionable Voltage Event Query 138
Sweep Direction Command 139
Sweep Direction Query 139
Sweep Frequency Start Command 140
Sweep Frequency Start Query 140
Sweep Spacing Command 140
Sweep Spacing Query 140
Sweep Time Command 141
Sweep Time Query 141
System Configure Query 141
System Error All Query 141
System Error Count Query 142
System Error Next Query 142
System Error Report Query 142
System Identify Command 143
System Identify Query 143
System Memory Query 143
System Memory Clear Command 143
System Restore Command 144
System Restore Query 144
System Temperature Query 144
System Test Count Query 144
System Test Report Query 145
System Undo Command 145
Trace Copy Query 146
Trace Invert Query 147
Trace Output Command 148
Trace Output Query 148
Trace Preamble Query 149
Trace Ready Query 150
Trace Reference Command 151
Trace Reference Query 151
Trace Scale Query 152
Trace Waveform Command 153
Trace Waveform Query 153
Trace Waveform Check Query 154
Trace Waveform Clear Command 154
Trace Waveform Clear All Command 154

Trace Waveform Points Query 155
Trace Waveform Valid Query 155
Trigger Delay Command 156
Trigger Delay Query 156
Trigger External Impedance Command 156
Trigger External Impedance Query 156
Trigger External Level Command 156
Trigger External Level Query 156
Trigger Internal Frequency Command 157
Trigger Internal Frequency Query 157
Trigger Pattern Mask Command 158
Trigger Pattern Mask Query 158
Trigger Pattern Truth Command 159
Trigger Pattern Truth Query 159
Trigger Slope Command 159
Trigger Slope Query 159
Trigger Source Command 160
Trigger Source Query 160
Trigger Timestamp Query 160
Voltage Amplitude Command 161
Voltage Amplitude Query 161
Voltage Offset Command 162
Voltage Offset Query 162
Waveform Maximum Query 162
Waveform Points Command 163
Waveform Points Query 163
Waveform Size Maximum Query 163
Waveform Size Minimum Query 163
Waveform Size Quantum Query 164
Waveform Switch Mode Command 164
Waveform Switch Mode Query 164
Width Command 165
Width Query 165

Specifications 166
Outputs 166
Digital-to-Analog Converter (DAC) 167
Operation Modes 168
Continuous Mode 168
Burst Mode 168
Binary Modulation Mode 168
Sweep Mode 168
Trigger 169
Arm 169
External Input 169
Sync Outputs 170
External Output 170
Event Outputs 170
Standard Functions 171
Sine 171
Square 171

Triangle 171
Ramp (sawtooth) 171
DC 171
Haversine 171
Havercosine 172
Half Cycle Sine 172
Pulse 172
Sinc Pulse 172
Gaussian Pulse 172
Lorentz Pulse 172
Noise 172
AM 173
FM 173
Multi-Tone 173
Serial Data 173
Arbitrary Waveforms 173
Waveform Sequences 173
Waveform Operations 174
Instrument Setup Storage 174
Status Reporting 174
PCI/PXI Data Interface 175
VXI Data Interface 175
LXI Data Interface 175
PXI XJ4 Trigger & Clock Pin Usage 176
PCI Timing Expansion Connector Pin Usage 176
VXIbus P2 Trigger & Clock Pin Usage 176
LED Indicators 177
Power 178
Physical 179
Temperature Range 179
Relative Humidity 179
Altitude 179
Safety 180
Electromagnetic Compatibility 180
CE Compliance 180
LXI Conformance 180

Default Reset Conditions 181

Error Codes 184

Commands Index 192
IEEE-488 Common Commands 192
SCPI Instrument Specific Commands 192

List of Figures

Figure 1.1: Photo of the ZT5210 VXI, PCI, PXI and LXI 16
Figure 1.2: ZT5211 PXI/PCI and ZT5212 VXI/LXI Front Panels 18
Figure 1.3: ZT5210 LXI Back Panel 20
Figure 2.1: Functional Block Diagram 22
Figure 2.2: Data Flow Diagram 23
Figure 2.3: DAC Clock Diagram 24
Figure 2.4: Waveform Sequencing 27
Figure 2.5: Arbitrary Waveform Discontinuity 29
Figure 2.6: Sine Wave 35
Figure 2.7: Square Wave 35
Figure 2.8: Triangle Wave 36
Figure 2.9: Ramp Up Wave 36
Figure 2.10: Ramp Down Wave 37
Figure 2.11: Half Cycle Sine Wave 38
Figure 2.12: Pulse Wave 39
Figure 2.13: Sinc Wave 39
Figure 2.14: Gaussian Wave 40
Figure 2.15: Lorentz Wave 40
Figure 2.16: AM Wave 41
Figure 2.17: FM Wave 42
Figure 2.18: Multi-Tone Wave (3 tones) 42
Figure 2.19: Serial Data Wave (8 bits, 01010010_2 word) 43
Figure 2.20: Trigger Initiate Model 45
Figure 2.21: Trigger Methods 45
Figure 2.22: Trigger Processing 46
Figure 2.23: ZT5210 Output Signal Conditioning 48
Figure 2.24: Attenuation due to Output Impedance 48
Figure 2.25: Output Voltage Maximum 49
Figure 2.26: Status Register 52
Figure 3.1: PCI Timing Expansion Connector 57
Figure 3.2: Driver Interface Layers 61

List of Tables

Table 1.1: Product Options 15
Table 1.2: Front Panel Connectors 17
Table 1.3: PCI/PXI/VXI Front Panel LEDs 19
Table 1.4: LXI Front Panel LEDs 19
Table 1.5: Back Panel Connectors 20
Table 1.6: Numeric Prefixes 21
Table 2.1: Number of Points in Standard Function 29
Table 2.2: Self-Test Errors 54
Table 3.1: PCI Timing Expansion Connector Pin Out 57
Table 3.2: LXI Ports 59
Table 3.3: LXI WTB Connector Pin Out 60
Table 4.1: Parameter Types 64

Introduction

Description

The ZT5210 instrument series consist of 14-bit, 200 million Samples-per-second (MS/s) Arbitrary Waveform Generators (AWGs). Each AWG instrument is built around a high speed, deep memory sequencer and embedded TMS320C6412 Digital Signal Processor (DSP). Together with the host processor and software, the instrument provides a simple powerful way to create and customize waveforms.

The AWG has two operation methods: it may act as a Function Generator, automatically producing standard waveform shapes with adjustable parameters, or it may act as an AWG, where the user may create custom waveforms and combine them through the sequencing functionality. Built-in DSP functions provide a variety of waveform manipulation resources. All AWG operations are controlled from an intuitive, software-based user interface running on the instrument bus host processor.

Product Options and Platforms

Model Number	Output Channels	Maximum Clock Rate	Standard Memory per Channel	Maximum Output Voltage	Supported Platforms
ZT5211	2	200 MS/s	32 MiSamples	±14 V	PCI/PXI/VXI/LXI
ZT5212	4	200 MS/s	32 MiSamples	±14 V	VXI/LXI

Table 1.1: Product Options

There are two instrument options housed within four modular instrument platforms. Two channel instruments are model number ZT5211 and four channel instruments are ZT5212. Table 1.1 shows the product options for the instrument that define channels, maximum clock rate and memory size. When referring to a specific option, this manual shall use the model number designations. When referring to all options within an instrument series, this manual shall use ZT5210. Functionality descriptions assume four channels; note that ZT5211 instruments do not have input channels 3 and 4 but are otherwise the same.

The M-Class includes AWGs built for four platforms: PCI, PXI, VXI and LXI. PCI and PXI are available only with two channels; VXI and LXI may be either two or four channel products. All instrument platforms have the same functionality with only a few exceptions. All platforms use the same software; see *Interface (Chapter 3)* for more detail on the different platform interfaces.

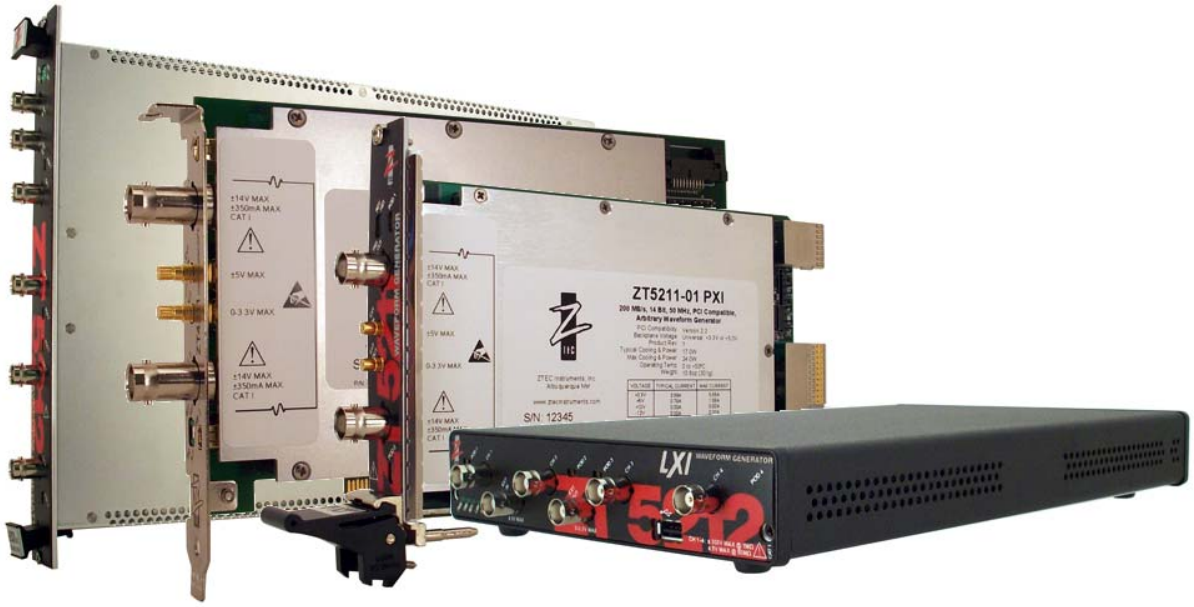


Figure 1.1: Photo of the ZT5210 VXI, PCI, PXI and LXI

Front Panel

Figure 1.2 shows the ZT5211 and ZT5212 front panels. Table 1.2 lists the front panel connectors.

Label	Description
CH 1	Channel 1 Output signal (BNC)
POD 1	Channel 1 ZTEC® accessory connector. This feature is not currently implemented.
CH 2	Channel 2 Output signal (BNC)
POD 2	Channel 2 ZTEC® accessory connector. This feature is not currently implemented.
CH 3	Channel 3 Output Signal (BNC)
POD 3	Channel 3 ZTEC® accessory connector. This feature is not currently implemented.
CH 4	Channel 4 Output signal (BNC)
POD 4	Channel 4 ZTEC® accessory connector. This feature is not currently implemented.
EXT IN	External Input (SMB PCI/PXI) (BNC VXI/LXI)
EXT OUT	External Output (SMB PCI/PXI) (BNC VXI/LAN)
USB Port	USB Port (LXI only). This feature is not currently implemented.

Table 1.2: Front Panel Connectors

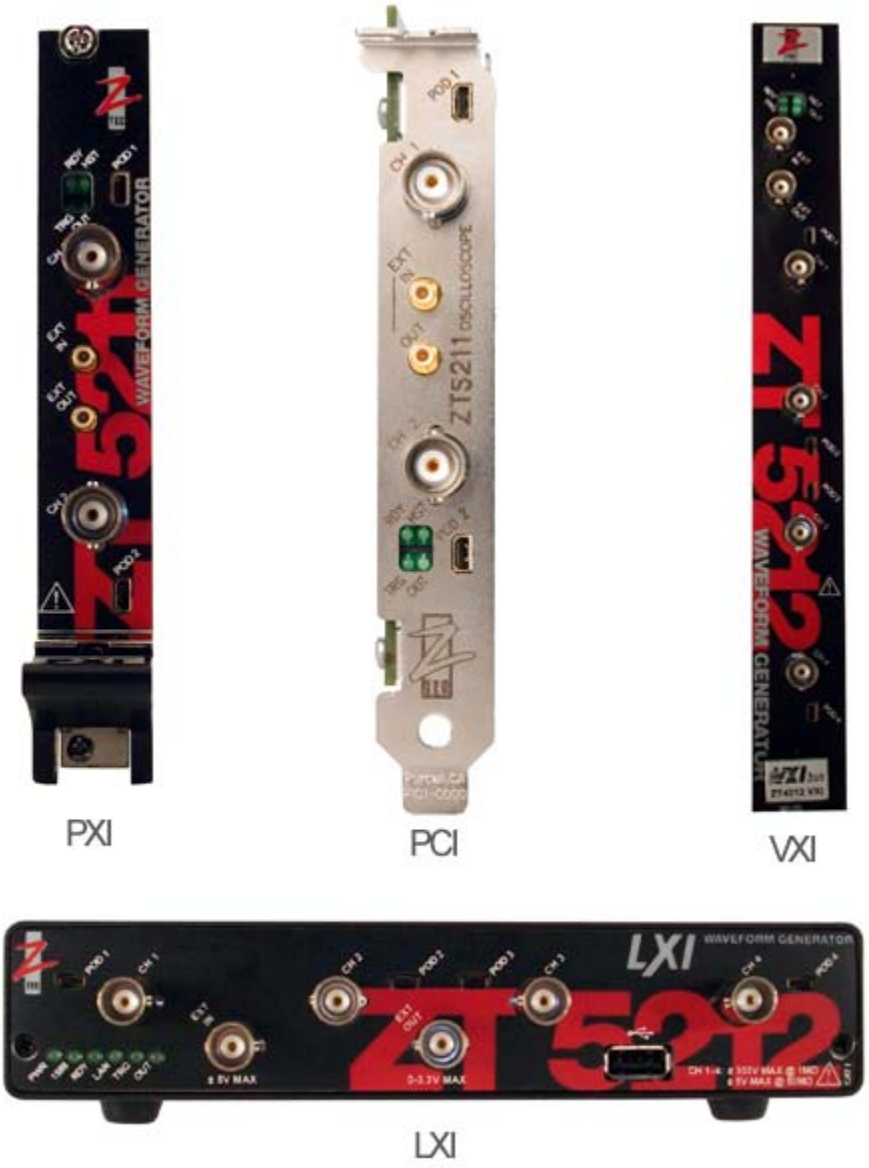


Figure 1.2: ZT5211 PXI/PCI and ZT5212 VXI/LXI Front Panels

The PCI/PXI and VXI platforms have four LED indicators on the front panel. Table 1.3 lists the LED states.

Label	Name	States
RDY	Ready	OFF: Hardware failure ON: Unit has passed power-up self-diagnostics TOGGLE: Unit has an error pending in error queue
HST	Host	OFF: Interface fault ON: Normal interface operation TOGGLE: Device identify command received
TRG	Trigger	OFF: Trigger event not detected ON/PULSE: Trigger complete event detected
ACT	Active	OFF: Instrument not generating waveforms ON/PULSE: Waveform generation initiated

Table 1.3: PCI/PXI/VXI Front Panel LEDs

The LXI platform has six LED indicators on the front panel. Table 1.4 lists the LED states.

Label	Name	States
PWR	Power	OFF: Hardware failure ON: Unit is powered
1588	1588 Clock Status	OFF: Currently not implemented
RDY	Ready	OFF: Hardware failure ON: Unit has passed power-up self-diagnostics TOGGLE: Unit has an error pending in error queue
LAN	Host	OFF: Interface fault ON: Normal interface operation TOGGLE: Device identify command received
TRG	Trigger	OFF: Trigger event not detected ON/PULSE: Trigger complete event detected
ACT	Active	OFF: Instrument not generating waveforms ON/PULSE: Waveform generation initiated

Table 1.4: LXI Front Panel LEDs

Back Panel

Figure 1.3 shows the ZT5210LXI back panel. Table 1.5 lists the back panel connectors in order from left to right.

Figure 1.3: ZT5210 LXI Back Panel

Label	Description
	Video Output / VGA Connector. Connect a monitor to display connection information.
	LAN over USB connector. This feature is not currently implemented.
	Ethernet LAN connector. Connect the device to the network with either a patch cable or a crossover cable. Supports 10/100 Mbps; does not support Auto-MDIX
LAN RST	Reset all LAN configuration settings.
WTB	Wired Trigger Bus
n/a - (no label)	AC power supply connection Supports automatic ranging. Voltage 90-264VAC and frequencies from 47-63Hz.

Table 1.5: Back Panel Connectors

Terminology

When referring to a specific option, this manual shall use the part number designations, possibly followed by a platform. Instrument series will be referred to by replacing some part number digits with a generic 'x'. When referring to all options within an instrument series, this manual shall use 'ZT521x'. When referring to all M-Class waveform generator part numbers, this manual shall use 'ZT5xxx' or 'M-Class AWG'. Functionality descriptions assume four channels; note the ZT5xx1 (2-channel M-Class AWG) instruments do not have output channels 3 and 4 but are otherwise the same.

Examples:

"ZT521x" or "ZT5210 Series" = ZT5211, ZT5212; all platforms

"ZT5xx1" = ZT5211; all platforms

When referring to large numeric values this manual will use SI (International System of Units) and IEC (International Electrotechnical Commision) standard prefixes. Common prefixes are listed in Table 1.6.

Prefix	Multiplier
n (nano)	1/(1000x1000x1000)
µ (micro)	1/(1000x1000)
m (mili)	1/1000
k (kilo)	1000
M (Mega)	1000x1000
G (Giga)	1000x1000x1000
Ki (Kibi)	1024
Mi (Mebi)	1024x1024
Gi (Gibi)	1024x1024x1024

Table 1.6: Numeric Prefixes

Additional Resources

ZTEC® Instruments, Inc. offers several hardware and software resources to use with the M-Class product line. Please visit the website www.ztecinstruments.com for the latest information and versions. Detailed information is also available in the individual manuals. Resources include: ZWave® M-Class soft front panel, ZFind™ Resource Manager, instrument drivers and cables. Please visit www.ztecinstruments.com/support for examples, FAQs, downloads, and customer support assistance.

Functionality and Operation

Functional Block Diagram

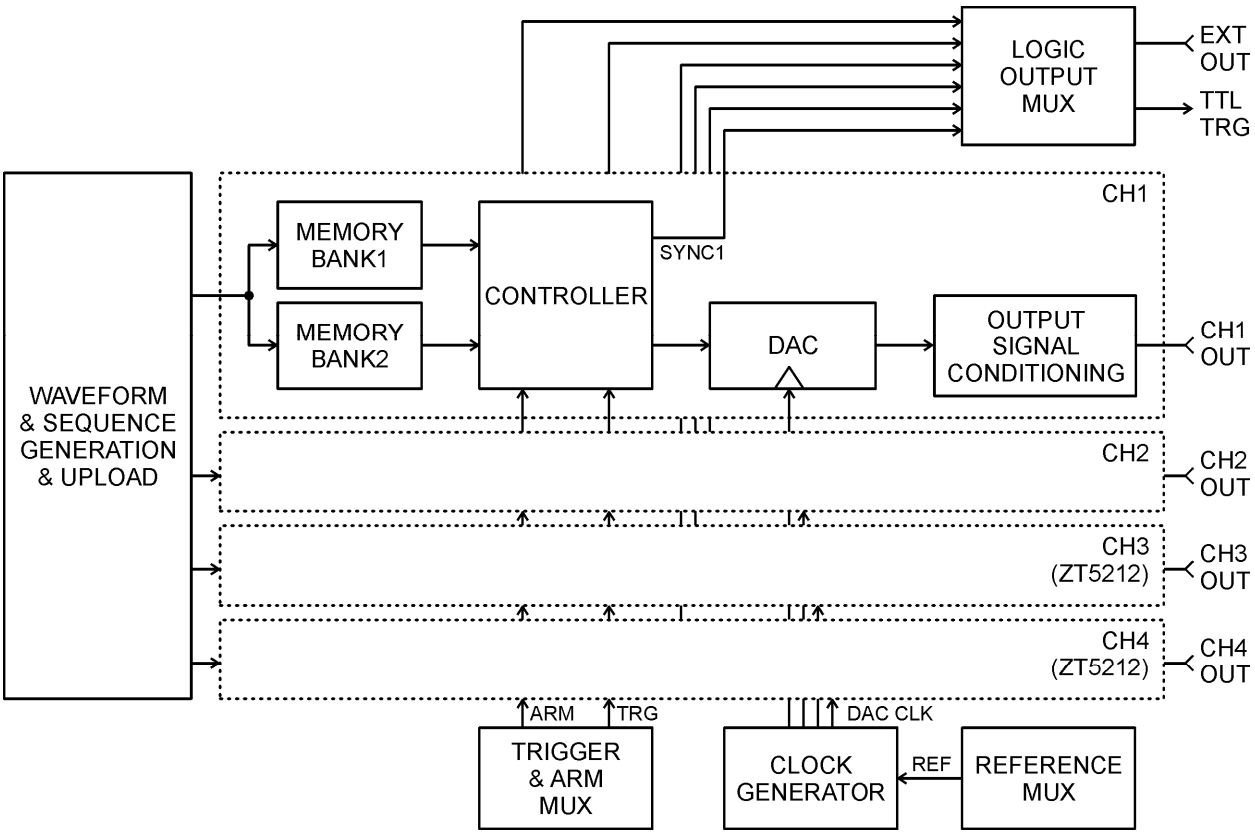


Figure 2.1: Functional Block Diagram

The M-Class Arbitrary Waveform Generator (AWG) has the following features (See Figure 2.1):

- The waveform generator and upload functionality creates standard and non-standard waveforms using an on-board digital signal processor (DSP).
- Each digital-to-analog converter (DAC) has two memory banks that can be used in a ping-pong fashion for simultaneous waveform upload and generation.
- The controller provides the waveform and sync generation controls for the dual memory banks.
- Each DAC converts its digital waveform data stream to an analog signal.
- The output signal conditioning provides the amplitude, offset and filter controls for the analog output signals.

- The reference multiplexer selects the time base reference 10 MHz clock.
- The clock generator creates four independent DAC clocks with flexible frequency controls.
- The trigger and arm multiplexer selects the trigger and arm sources for triggered, gated or modulated waveform generation.
- The logic output multiplexer routes sync and trigger sources to the front-panel and backplane logic outputs.

Data Flow

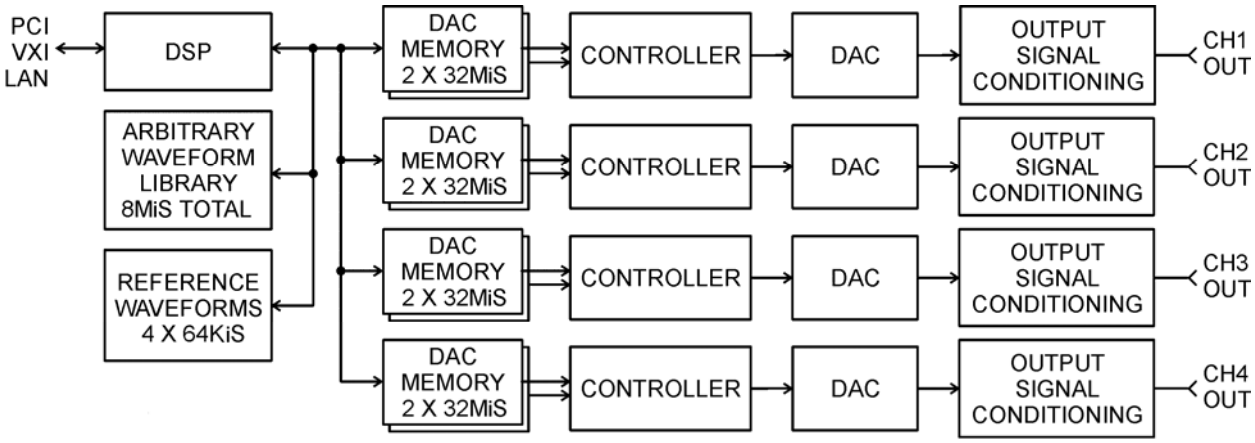


Figure 2.2: Data Flow Diagram

The data flow of the M-Class AWG is shown in Figure 2.2. A digital signal processor (DSP) provides the on-board intelligence for command communication, waveform generation, sequence generation, supervisory control, and all other embedded processing functionality. The host communicates to the instrument DSP via that platform's specific interface: PCI, PXI, VXI or LAN. In addition to sending commands and queries, the host can upload waveform data for arbitrary waveform generation. Each output channel has dual DAC memory buffers of up to 32 MiSamples (64 MiB) in depth. While one buffer feeds a data stream to its DAC for conversion to the analog output, the other buffer can be loaded with the next waveform to be generated. This dual ping-pong buffer architecture enables instantaneous, seamless waveform switching upon command, trigger event or modulation input state.

Each DAC memory buffer can be loaded with standard functions generated by the DSP, arbitrary waveforms uploaded from the host, or waveform sequences that piece together arbitrary waveforms in stages to create compound waves. When generating waveform sequences, the DSP uses the arbitrary waveform library or the four reference waveforms. The arbitrary waveform library provides up to 8 MiSamples (16 MiB) in total memory that can be loaded with up to 4096 arbitrary waveforms. The four reference waveforms, REF1–4, are each up to 32 KiSamples (64 KiB) in length, and use non-volatile memory that is maintained when the unit is powered off.

Clock Generation and Synchronization

DAC Sampling Clock

The instrument supports flexible digital-to-analog converter (DAC) sampling clock configurations. All internal sampling clocks are synchronized by a phase-locked loop (PLL) that is locked to the 10 MHz time base reference. Each individual DAC clock is generated with a separate direct digital synthesizer (DDS) and clock divider that provides greater than nine digits of frequency resolution (32-bit or 1 part in 4.2 billion) between 200 Hz and 200 MHz. When generating arbitrary waveforms or waveform sequences, each DAC on the ZT5210 can use its independent clock or the common clock source. The common clock source is user-selected from any one of the two (ZT5211) or four (ZT5212) DAC clocks. A common clock source is useful when synchronizing multiple output channels to the same DAC clock frequency. Note that standard functions always use independent clock sources.

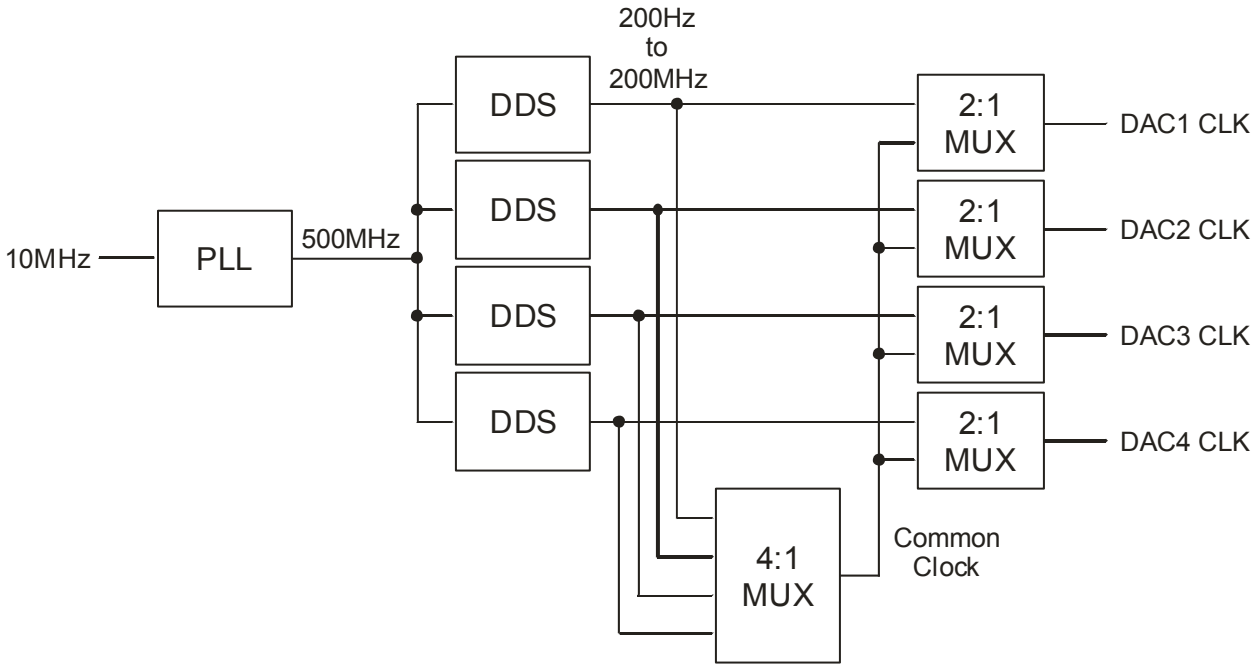


Figure 2.3: DAC Clock Diagram

Time Base Reference Clock

The instrument supports flexible time base reference configurations. The 10 MHz time base reference is used to synchronize all internal timing including the sampling clock for the DACs. The source of the time base reference is selectable between an internal temperature-compensated crystal oscillator (TCXO) the backplane CLK10 reference signal, and the external reference (EXT IN). The time base frequency must be 10 MHz ±100 ppm. The internal TCXO reference provides ±2.5 ppm frequency accuracy. The reference oscillator source is selected using the *Reference Oscillator Source Command*.

Waveform Types

There are three output signal types available when using the M-Class AWG: standard functions, arbitrary waveforms and waveform sequences. The output signal type can be independently selected for each channel, allowing a combination of standard functions and arbitrary waveforms to be generated simultaneously on the different output channels. The *Output Mode Command* is used to select the waveform type for an output channel. The output signal changes without discontinuity at the end of a waveform cycle when changing the function shape or arbitrary waveform data. In contrast, the output signal changes instantaneously when altering the vertical settings (output amplitude or offset), the function frequency or the arbitrary waveform clock frequency.

Standard functions are generated internally by the instrument, and allow the user to easily specify waveform parameters to create a number of pre-defined waveform shapes. Arbitrary waveforms are created by the user and uploaded to the instrument memory. With arbitrary waveforms, the DAC data stream is entirely user-defined and provides the flexibility to create any arbitrary analog output signal. Waveform sequences piece together arbitrary waveforms in stages to create compound waves. Note that the *Waveform Copy* command provides a method to copy standard functions to instrument memory for use in arbitrary waveform sequences.

Standard Functions

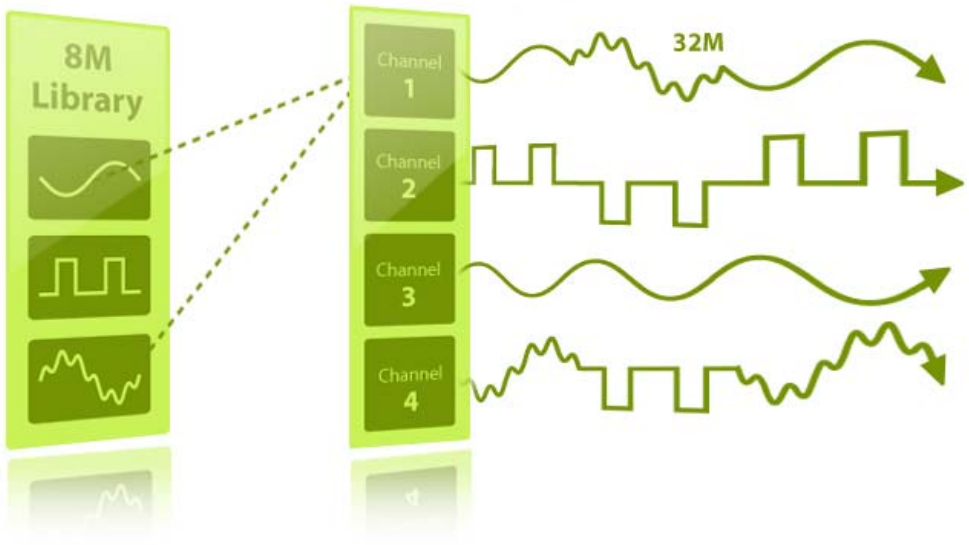
Standard functions are generated internally by the instrument to create pre-defined waveform shapes such as sine, square, triangle, ramp, etc. Programmable waveform parameters that are common to almost every standard function include amplitude, offset, frequency and phase. The amplitude and offset parameters control the vertical voltage swing, and are independently programmable for each output channel. The function frequency parameter controls the function repetition period for one cycle of the waveform. The phase parameter controls the relative phase between each output channel. A phase setting of zero degrees corresponds to the standard function starting and stopping at the zero DAC code value. A constant phase setting for a channel will ensure continuity when switching between different shape waveforms.

Arbitrary Waveforms

Arbitrary waveforms can be uploaded to the instrument memory and routed directly to the DAC to create arbitrary analog output signals. The entire 32 MiSample DAC memory is available to load a single, non-sequenced arbitrary waveform. The length of an arbitrary waveform can vary between 4 samples and 32 MiSamples. The DAC clock rate sets the time interval at which each data point is converted from digital data to an analog signal. The DAC clock rate is programmable between 200 S/s and 200 MS/s. The waveform size and clock rate parameters are independently programmable for each output channel. In addition, the vertical parameters for amplitude and offset are also independently programmable for each output channel.

Waveform Sequences

The M-Class Arbitrary Waveform Generators support the generation of a sequence of waveforms. Up to 8 sequences can be defined by the user. Waveform sequences piece together arbitrary waveforms in stages to create compound waves. The arbitrary waveform library and the reference waveforms contain the arbitrary waveform data that are used in the stages of the waveform sequence.



The *Waveform Copy* command allows standard functions to be copied to the arbitrary waveform library or to reference channels for use in a sequence. Arbitrary waveforms can be looped and reused in a sequence to provide additional waveform generation flexibility for long waveform sequences. Each waveform in the sequence is repeated a discrete number of times before proceeding to the next waveform. The repeat or loop count for each waveform in the sequence is user selectable between 1 and 65535. Each set of repeated waveforms is referred to as a waveform stage. Figure 2.4 depicts an example of a waveform sequence with three waveform stages. The number of waveform stages in a sequence is user selectable between 2 and 4096.

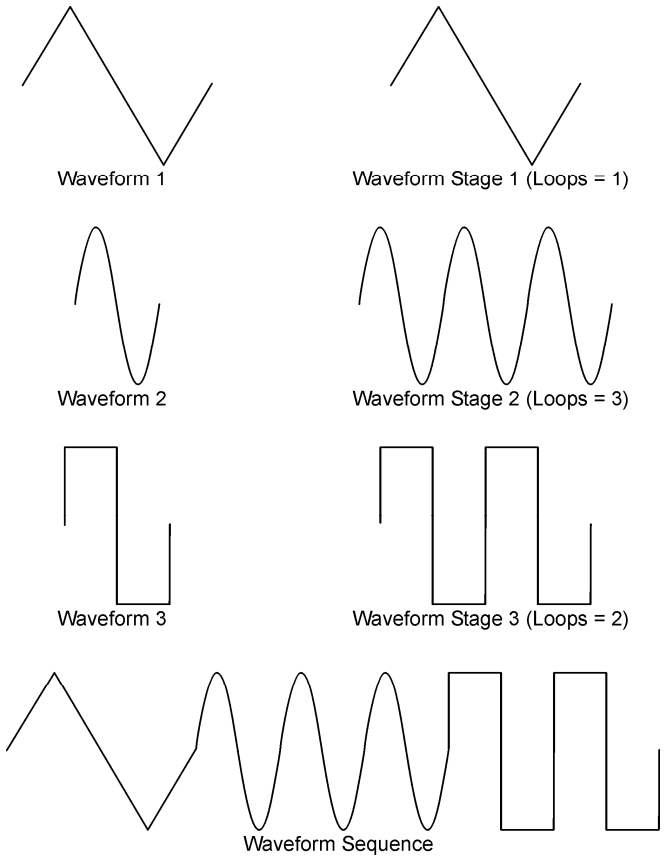


Figure 2.4: Waveform Sequencing

Each output channel may be configured independently for sequence generation, and each output channel may use a unique waveform sequence. For each output channel, the maximum amplitude, offset and DAC sample rate must be the same for all waveform stages in the sequence. The waveform size and loop count are uniquely defined for each waveform stage in the sequence.

Waveform Memory

Waveform Size and Cycle Period

The waveform length in samples and the DAC sampling clock rate in samples per second (S/s) define the corresponding period for one cycle of waveform data. The waveform length is user selectable between 4 Sample and 32 MiSamples using the *Waveform Points* command. The DAC sampling clock rate is user selectable between 200 S/s and 200 MS/s with greater than nine digits of frequency resolution. Waveform length, DAC sampling clock rate, and the repetition period for one waveform cycle are related by the following equation:

Waveform cycle period = sample points/sample rate

Standard Function Size

Standard functions are generated as one period, or cycle, in memory. By default, one cycle of a standard function is generated as an N point waveform, where N is shown in the following table. The number of points in a standard function, N, can be manually selected using the *Waveform Points* command. With the N-point waveform, the DAC sampling clock rate is set to N times the standard function frequency.

Function Frequency (f)	Function Period (per)	Number of Points (N)	Sample Rate
40 MHz < f ≤ 50 MHz	25 ns > per ≥ 20 ns	4	160 MS/s < r ≤ 200 MS/s
33 MHz < f ≤ 40 MHz	30 ns > per ≥ 25 ns	5	166 MS/s < r ≤ 200 MS/s
25 MHz < f ≤ 33 MHz	40 ns > per ≥ 30 ns	6	150 MS/s < r ≤ 200MS/s
20 MHz < f ≤ 25 MHz	50 ns > per ≥ 40 ns	8	160 MS/s < r ≤ 200 MS/s
12.5 MHz < f ≤ 20 MHz	80 ns > per ≥ 50 ns	10	125 MS/s < r ≤ 200MS/s
10 MHz < f ≤ 12.5 MHz	100 ns > per ≥ 80 ns	16	160 MS/s < r ≤ 200 MS/s
6.25 MHz < f ≤ 10 MHz	160 ns > per ≥ 100 ns	20	125 MS/s < r ≤ 200 MS/s
5 MHz < f ≤ 6.25 MHz	200 ns > per ≥ 160 ns	32	160 MS/s < r ≤ 200MS/s
4 MHz < f ≤ 5 MHz	250 ns > per ≥ 200 ns	40	160 MS/s < r ≤ 200 MS/s
2 MHz < f ≤ 4 MHz	500 ns > per ≥ 250 ns	50	100 MS/s < r ≤ 200 MS/s
1 MHz < f ≤ 2 MHz	1 µs > per ≥ 500 ns	100	100 MS/s < r ≤ 200 MS/s
500 kHz < f ≤ 1 MHz	2 µs > per ≥ 1 µs	200	100 MS/s < r ≤ 200 MS/s
250 kHz < f ≤ 500 kHz	4 µs > per ≥ 2 µs	400	100 MS/s < r ≤ 200 MS/s
200 kHz < f ≤ 250 kHz	5 µs > per ≥ 4 µs	500	100 MS/s < r ≤ 200 MS/s
100 kHz < f ≤ 200 kHz	10 µs > per ≥ 5 µs	1000	100 MS/s < r ≤ 200 MS/s
50 kHz < f ≤ 100 kHz	20 µs > per ≥ 10 µs	2000	100 MS/s < r ≤ 200 MS/s
40 kHz < f ≤ 50 kHz	25 µs > per ≥ 20 µs	4000	160 MS/s < r ≤ 200 MS/s
20 kHz < f ≤ 40 kHz	50 µs > per ≥ 25 µs	5000	100 MS/s < r ≤ 200 MS/s
10 kHz < f ≤ 20 kHz	100 µs > per ≥ 50 µs	10000	100 MS/s < r ≤ 200 MS/s
5 kHz < f ≤ 10 kHz	200 µs > per ≥ 100 µs	20000	100 MS/s < r ≤ 200 MS/s
4 kHz < f ≤ 5 kHz	250 µs > per ≥ 200 µs	40000	160 MS/s < r ≤ 200 MS/s
2 kHz < f ≤ 4 kHz	500 µs > per ≥ 250 µs	50000	100 MS/s < r ≤ 200 MS/s
1 kHz < f ≤ 2 kHz	1 ms > per ≥ 500 µs	10000	10 MS/s < r ≤ 20 MS/s
500 Hz < f ≤ 1 kHz	2 ms > per ≥ 1 ms	20000	10 MS/s < r ≤ 20 MS/s
400 Hz < f ≤ 500 Hz	2.5 ms > per ≥ 2 ms	40000	16MS/s < r ≤ 20 MS/s
200 Hz < f ≤ 400 Hz	5 ms > per ≥ 2.5 ms	50000	10 MS/s < r ≤ 20 MS/s
100 Hz < f ≤ 200 Hz	10 ms > per ≥ 5 ms	10000	1 MS/s < r ≤ 2 MS/s
50 Hz < f ≤ 100 Hz	20 ms > per ≥ 10 ms	20000	1 MS/s < r ≤ 2 MS/s
40 Hz < f ≤ 50 Hz	25 ms > per ≥ 20 ms	40000	1.6 MS/s < r ≤ 2 MS/s
20 Hz < f ≤ 40 Hz	50 ms > per ≥ 25 ms	50000	1 MS/s < r ≤ 2 MS/s
10 Hz < f ≤ 20 Hz	100 ms > per ≥ 50 ms	10000	100 kS/s < r ≤ 200 kS/s
5 Hz < f ≤ 10 Hz	200 ms > per ≥ 100 ms	20000	100 kS/s < r ≤ 200 kS/s
4 Hz < f ≤ 5 Hz	250 ms > per ≥ 200 ms	40000	160 kS/s < r ≤ 200 kS/s

Table 2.1: Number of Points in Standard Function

Function Frequency (f)	Function Period (per)	Number of Points (N)	Sample Rate
4 Hz < f ≤ 5 Hz	250 ms > per ≥ 200 ms	40000	160 kS/s < r ≤ 200 kS/s
2 Hz < f ≤ 4 Hz	500 ms > per ≥ 250 ms	50000	100 kS/s < r ≤ 200 kS/s
1 Hz < f ≤ 2 Hz	1 s > per ≥ 500 ms	10000	10 kS/s < r ≤ 20 kS/s
0.5 Hz < f ≤ 1 Hz	2 s > per ≥ 1 s	20000	10 kS/s < r ≤ 20 kS/s
0.4 Hz < f ≤ 0.5 Hz	2.5 s > per ≥ 2 s	40000	16 kS/s < r ≤ 20 kS/s
0.2 Hz < f ≤ 0.4 Hz	5 s > per ≥ 2.5 s	50000	10 kS/s < r ≤ 20 kS/s
0.1 Hz < f ≤ 0.2 Hz	10 s > per ≥ 5 s	10000	1 kS/s < r ≤ 2 kS/s
0.05 Hz < f ≤ 0.1 Hz	20 s > per ≥ 10 s	20000	1 kS/s < r ≤ 2 kS/s
0.04 Hz < f ≤ 0.05 Hz	25 s > per ≥ 20 s	40000	1.6 kS/s < r ≤ 2 kS/s
0.02 Hz < f ≤ 0.04 Hz	50 s > per ≥ 25 s	50000	1 kS/s < r ≤ 2 kS/s
0.01 Hz < f ≤ 0.02 Hz	100 s > per ≥ 50 s	20000	200 S/s < r ≤ 400 S/s
0.005 Hz < f ≤ 0.01 Hz	200 s > per ≥ 100 s	40000	200 S/s < r ≤ 400 S/s
0.004 Hz < f ≤ 0.005 Hz	250 s > per ≥ 200 s	50000	200 S/s < r ≤ 400 S/s
0.002 Hz < f ≤ 0.004 Hz	500 s > per ≥ 250 s	100000	200 S/s < r ≤ 250 S/s
0.001 Hz ≤ f ≤ 0.002 Hz	1000 s ≥ per ≥ 500 s	200000	200 S/s ≤ r ≤ 400 S/s

Table 2.1: Number of Points in Standard Function

Arbitrary Waveform Size

Arbitrary waveforms and arbitrary waveform sequences are loaded as one cycle in memory and may vary between 4 Sample and 32 MiSamples. The DAC sampling rate is user programmable and defines the waveform cycle period (the time duration at which one cycle of the arbitrary waveform or waveform sequence repeats). When generating arbitrary waveforms and waveform sequences, the wrap condition of the waveform should be considered to ensure that there is continuity in the waveform. Figure 2.5 shows a non continuous periodic cycle in waveform memory that causes a waveform wrap discontinuity.

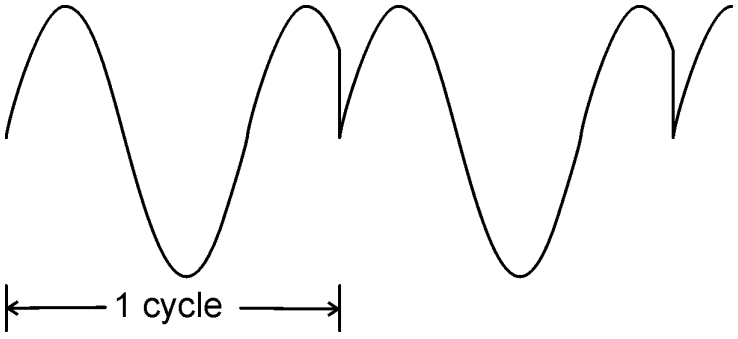
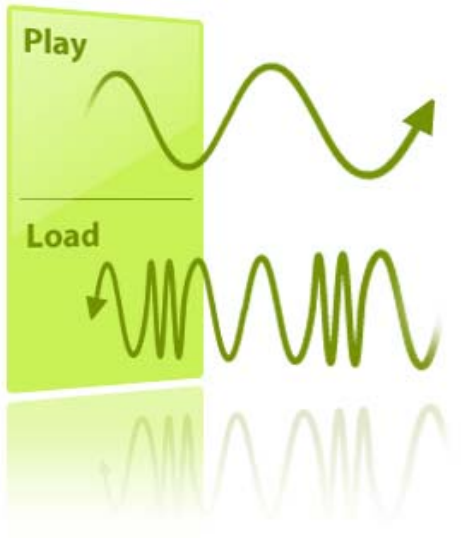


Figure 2.5: Arbitrary Waveform Discontinuity

Dual DAC Memory Banks

The DAC memory for each channel is divided into two separate memory banks to allow for simultaneous load and play of the output function, waveform or sequence. In all modes of operation other than binary modulation, the next desired output waveform may be loaded into memory while the present waveform is playing.

In binary modulation mode, waveform generation must be aborted before a new pair of binary modulation waveforms can be loaded into memory. Each of the pair of DAC memory banks is 32 MiSamples in depth. While one buffer feeds a data stream to its DAC for conversion to the analog output, the other buffer can be loaded with the next waveform to be generated. This dual ping-pong buffer architecture enables seamless or instantaneous waveform switching. The switching between the two banks can be controlled by host command or by the modulation input state. When using the binary modulation mode, the state of the modulation input source controls which of the two banks is used for generating the output.



Waveform Switching

The dual memory bank architecture allows the user to select seamless or instantaneous waveform switching for each output channel using the W*aveform Switch Mode Command*. Waveform switching refers to the manner in which the output waveform changes to a new waveform upon command or upon the modulation input state changes in Binary Modulation mode. Seamless switching eliminates transients or discontinuities in the waveform by completing a full cycle of the present waveform before switching to the new waveform. Instantaneous switching provides immediate waveform transition without waiting for the present cycle to complete. Instantaneous waveform switching will likely cause transients in the output waveform due to a discontinuity between the current and new waveforms being generated.

Arbitrary Waveform Library

When generating sequences of arbitrary waveforms, the DSP uses the arbitrary waveform library and the reference waveforms. The arbitrary waveform library consists of 8 MiSamples (16 MiB) of the DSP's SDRAM memory. The four reference waveforms, REF1–4, are each up to 32 KiSamples (64 KiB) in length. The arbitrary waveform library provides waveforms that are preloaded into memory. Once loaded, there waveforms can be referred to multiple times within a sequence or referred to by multiple channels.

The 8 MiSample SDRAM memory that is allocated for the waveform library is not prearranged or reserved for specific waveform data. Each waveform can be loaded in memory consecutively and is referenced by its start address and length. Waveforms in the library may be different lengths and there may be as many as 4096 waveforms in the library, as long as the total memory usage does not exceed 8 MiSamples (16 MiB). The waveform library allocates memory to waveforms in blocks of 256 Samples (512 B), so odd sized waveforms may not be able to use the full 8 MiSamples. When uploading waveforms into the waveform library, use the *Trace Waveform Check Query* to confirm that there is sufficient room in memory and to have a handle assigned for the data. Waveforms may be cleared from the memory independently to make room for other waveforms.

Reference Waveforms

The instrument can store up to 4 reference waveforms as part of the arbitrary waveform library for use within sequences. The reference waveforms, REF1–4, are stored in non-volatile Flash memory and are maintained when the unit is powered off. Reference waveforms are each limited to sizes of 32 KiSamples (64 KiB) or less.

Waveform Handles

Each waveform in the arbitrary waveform library is assigned a unique handle that is used as an identifier for generating waveform sequences. Up to 4096 waveform handles may be assigned within the 8 MiSample arbitrary waveform library. Each handle is associated with the memory address and length of the arbitrary waveform. The sequence table uses the waveform handles when defining the sequence order and loop count for each waveform stage in the sequence. Consequently, a waveform handle may appear multiple times within a sequence or within multiple channels, allowing the pre-loaded arbitrary waveforms to be reused.

Waveform Data Format

Each waveform is represented in memory as 16-bit signed integer codes that are fed to the DAC for conversion to an analog signal. There are two numeric data formats that are used for uploading or downloading waveform data to/from memory: 16-bit signed integers and 32-bit floating point real numbers. The *Format Data Command* is used to specify the upload/download data format that is used for all subsequent transfers. When using 16-bit signed integer format, the DAC codes are represented as numbers between -32,767 and +32,767. The two lowest bits (LSBs) of the 16-bit data are ignored by the 14-bit DAC on the ZT5210. When using 32-bit real format, the DAC codes are represented as normalized values between -1.0 and +1.0. The DSP processor converts the floating point numbers to signed integer DAC codes before loading the data into memory. In addition, both Intel (LSB) and Motorola (MSB) byte orders are supported and can be specified using the *Format Byte Order* command.

Waveform Operations

Waveform operations apply to all waveforms in memory on the instrument including: (a) DAC memory, (b) arbitrary waveform library, or (c) reference waveforms. Waveform operations allow the user to write, read, copy, invert, or scale waveform data in memory as follows:

- Upload: host loads waveform data for signal generation.
- Download: host reads waveform data for verification.
- Copy: copies waveform data between two locations in instrument waveform memory.
- Invert: inverts (2s complement) digital codes for waveform data.
- Scale: scales digital codes of waveform data to less than full-scale.

Waveform Upload

The instrument uses shared memory for the DAC ping-pong buffers and the arbitrary waveform library. The instrument provides high-speed direct memory access for all waveform upload operations. A single arbitrary waveform can be uploaded to the 32 MiSample DAC ping-pong memory buffer for any output channel. The ping-pong buffer provides upload access when the

instrument is simultaneously generating an output waveform. An arbitrary waveform can also be uploaded to the 8 MiSample arbitrary waveform library or to any reference waveform channel. The uploaded waveforms in the arbitrary waveform library and reference channels are available for use in a stage of a sequence for any output channel. The arbitrary waveform library must be preloaded prior to the generation of waveform sequences.

Waveform upload is a two step process; first, the waveform must be uploaded to the instrument's block buffer, and then moved to the selected channel or library location. There are several methods available for waveform upload. The simplest is to use the provided instrument driver function, which does everything in a single command. It is also possible to upload directly using buffer information provided by the zbind functions zbind_block_buffer_addr() and zbind_block_buffer_size(). With this information, the waveform can be uploaded to the buffer with zbind_blkout() and then moved to the channel or library location with *Trace Output Command, Trace Reference Command* or *Trace Waveform Command.*

Waveform Download

Waveform download allows the user to verify waveform data for standard functions, arbitrary waveforms, or waveform sequences. Waveform download is a two step process; first, the waveform must be moved from the selected channel or library location to a buffer, then the waveform must be downloaded from the instrument. The simplest way to download is to use the provided instrument driver command, which does everything in a single command. It is also possible to download directly through the zbind interface. *Trace Output Query*, *Trace Reference Query* or *Trace Waveform Query* will move the selected waveform to the buffer and return the address location and length of the waveform data for download. Once the waveform is in the buffer it can be downloaded in pieces of up to zbind_block_buffer_size() using zbind_blkin().

Waveform Copy

Waveform copy allows the user to duplicate waveform data from one section of memory to another. For example, this provides a mechanism to store waveforms to a reference channel, recall waveforms from a reference channel, or copy standard functions in DAC memory to the arbitrary waveform library. Copying moves waveform codes only it does not preserve any scaling information like voltage amplitude or frequency.

Waveform Invert

The *Trace Invert Query* provides a simple mechanism to invert the digital waveform data using 2s complement of the selected channel. This enables the creation of complementary or differential output signals.

Waveform Scale

The *Trace Scale Query* provides a simple mechanism to scale the digital waveform data of the selected channel. This allows the output signal to be scaled to less than full amplitude without affecting the maximum amplitude setting. This enables the relative adjustment of multiple waveforms that are played through the same signal conditioning, such as stages in a sequence or binary modulation waveforms.

Operation Modes

The operating mode for each channel can be selected independently using the *Operation Mode Command*. This allows different channels on an instrument to be operated in continuous, burst, binary modulation, or sweep modes simultaneously. One exception is that burst and binary modulation modes may not be used simultaneously on different channels. Also, when using burst mode on multiple channels, each channel must use the same trigger source. When using binary modulation mode on multiple channels, each channel must use the same binary modulation source.

Continuous Mode

Continuous mode is the default operation mode where the output signals are generated continuously when enabled. The instrument starts signal generation when initiated and stops signal generation when aborted.

Burst Mode

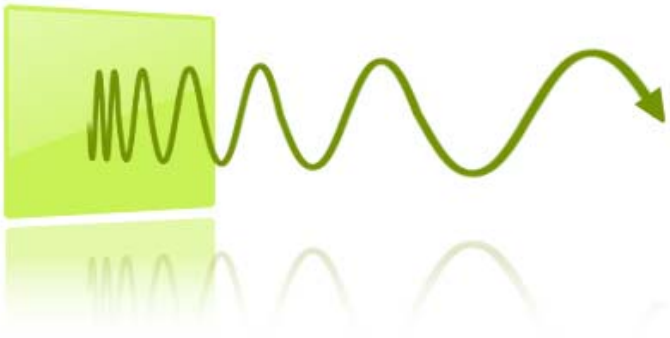
In burst mode, the instrument generates a discrete number of cycles upon a trigger event. The trigger event may be qualified by the arm condition if desired. The number of cycles to generate is programmable between 1 and 65,535 (2^{16}-1). For standard functions and arbitrary waveforms, a cycle is one period of the waveform. For arbitrary sequences, a cycle is one complete progression through all the waveforms in the sequence. Burst mode will either accept a single trigger event or continuously trigger based on the *Initiate Continuous Command*.

Binary Modulation Mode

Binary modulation mode allows an external or internal modulation source to be applied to switch an output channel between two preloaded waveforms. This functionality enables amplitude shift keying, frequency shift keying, phase shift keying, toggling between two arbitrary waveforms, or gated signal generation. For each of these applications, waveforms for both states are preloaded and the output waveform is determined by the state of the modulation source. The maximum amplitude, offset and DAC sample rate is the same for both binary state waveforms. The modulation source is selectable between the external input, TTL triggers 0-7, and ECL triggers 0-1 (VXI), star trigger (PXI/PCI), pattern trigger, or software control.

Sweep Mode

Sweep mode allows a swept-frequency output signal. When using sweep mode, the waveform shape in DAC memory is constant, and the DAC clock frequency is swept in a linear or logarithmic fashion. The start and stop frequencies are programmable, along with the sweep time duration from start to stop. The maximum sweep frequency range is three decades, which allows a maximum start-to-stop frequency ratio of 1000:1. Sweep modes of up, down or up and down are selectable. Sweep time durations of 1 µs to 100 seconds are supported. Note the sweep duration is common to all channels enabled in sweep mode.



Standard Function Descriptions

The following descriptions provide details for each of the standard functions that are internally generated by the instrument. A standard function is generated as a single cycle in memory as shown in the following figures. Most standard functions have user programmable amplitude, offset, frequency and phase. The function period is equal to the reciprocal of the function frequency. The phase sets the relative start and stop positions in the waveform cycle. For pulse and square functions, the user programmable duty cycle sets the positive width of the function.

Sine

A *Sine* function is a sinusoidal waveform defined by amplitude, offset, frequency, period, and phase. See figure 2.6.

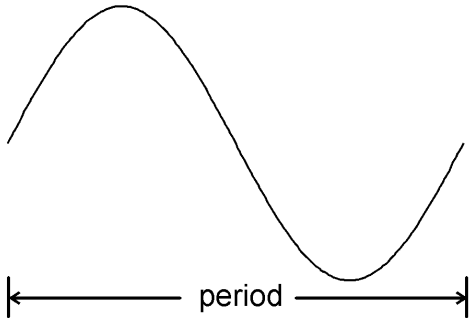


Figure 2.6: Sine Wave

Square

A *Square* function is a binary waveform that toggles between two levels. Edge transitions are immediate and limited only by the analog bandwidth and clock cycle time of the instrument. The user programmable duty cycle sets the positive width of the *Square* function. See the *Pulse* waveform for more flexible rise and fall time. Square Waves are defined by amplitude, offset, frequency, period, phase, positive width and duty cycle. See figure 2.7.

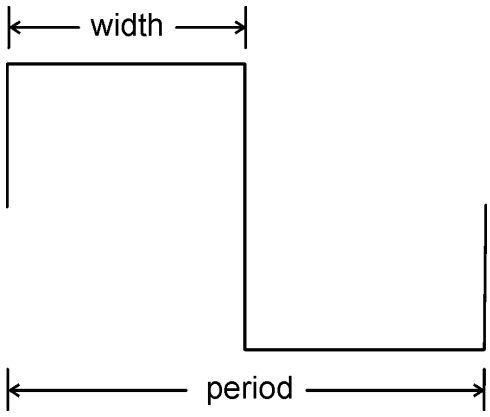


Figure 2.7: Square Wave

Triangle

A *Triangle* function ramps up and down creating a triangle-shape waveform as defined by the amplitude, offset, frequency, period and phase. See figure 2.8.

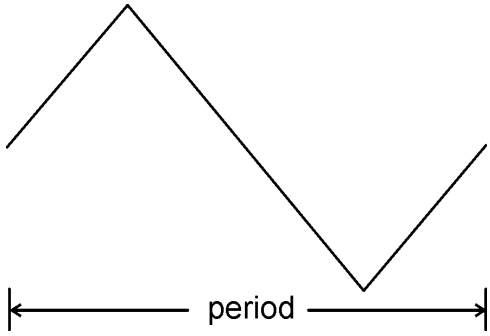


Figure 2.8: Triangle Wave

Ramp Up

A *Ramp Up* function is a sawtooth waveform that ramps up. Ramp functions are defined by amplitude, offset, frequency, period and phase. See figure 2.9.

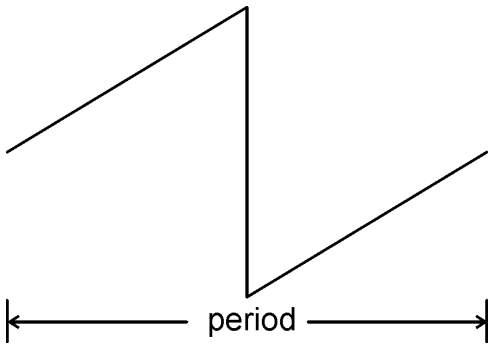


Figure 2.9: Ramp Up Wave

Ramp Down

A *Ramp Down* function is a sawtooth waveform that ramps down. Ramp functions are defined by amplitude, offset, frequency, period and phase. See figure 2.10.

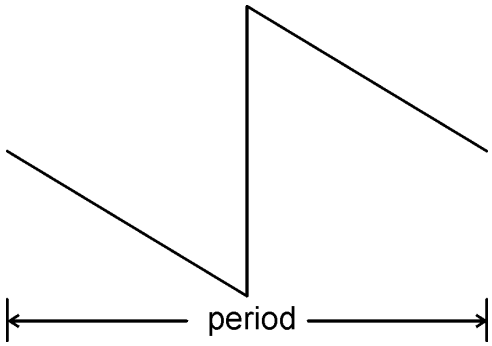


Figure 2.10: Ramp Down Wave

DC

A *DC* function is a constant value waveform set by the offset value. The DC function is defined only by the channel's offset.

Haversine

A *Haversine* function is half the versed sine waveform that equates to a sinusoid with a DC offset equal to half the peak-to-peak amplitude. This results in a waveform voltage swing between 0.0 Volts and the positive peak-to-peak voltage. A *Haversine* function (or haversed sine) is a trigonometric function defined as:

$$hav(t) = \tfrac{1}{2}[1- \cos(t)]$$

Haversine waveforms are defined by amplitude, offset, frequency, period and phase.

Havercosine

A *Havercosine* function is half the versed cosine waveform that equates to a sinusoid with a DC offset equal to half the peak-to-peak amplitude. This results in a waveform voltage swing between 0.0 Volts and the positive peak-to-peak voltage. A *Havercosine* function (or haversed cosine) is a trigonometric function defined as:

$$\text{cohav}(t) = \tfrac{1}{2}[1 - \sin(t)]$$

Havercosine waveforms are defined by amplitude, offset, frequency, period and phase.

Half Cycle Sine

A *Half Cycle Sine* function is one-half of a sinusoidal waveform. Half Cycle Sine waveforms are defined by amplitude, offset, frequency, period and phase. See figure 2.11.

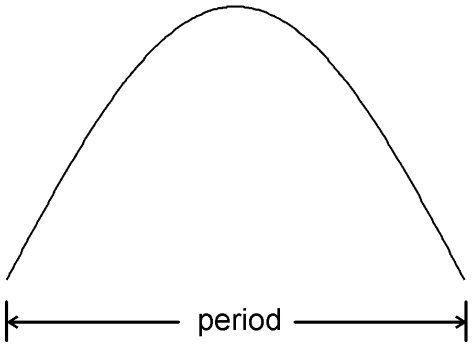


Figure 2.11: Half Cycle Sine Wave

Pulse

A *Pulse* function is a flexible square waveform with variable leading and trailing transition times (rise and fall time). The width is user programmable and adjusts the positive width of the *Pulse* function. Pulse waveforms are defined using amplitude, offset, frequency, period, leading transition time, trailing transition time, positive width and duty cycle. See figure 2.12.

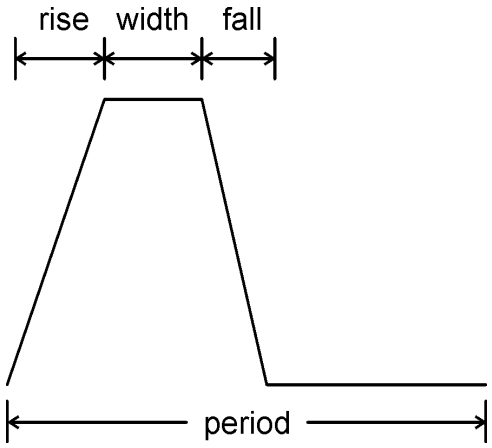


Figure 2.12: Pulse Wave

Sinc Pulse

A *Sinc Pulse* function is a limited bandwidth impulse waveform with variable sinc frequency. See figure 2.13. A *Sinc Pulse* function is expressed by the following equation, where f is the sinc frequency in Hz:

$$sinc(t) = \sin(2\pi f*t)/(2\pi f*t)$$

Sinc pulses are defined by amplitude, offset, frequency, period and sinc frequency.

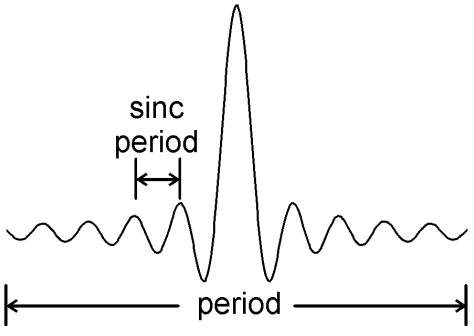


Figure 2.13: Sinc Wave

Gaussian Pulse

A *Gaussian Pulse* function is standard normal distribution or bell-shaped curve with variable standard deviation. See figure 2.14. A *Gaussian Pulse* function is expressed by the following equation, where σ is the standard deviation:

$$gauss(t) = \exp[-(t/\sigma)^2]$$

Gaussian pulses are defined by amplitude, offset, frequency, period and standard deviation.

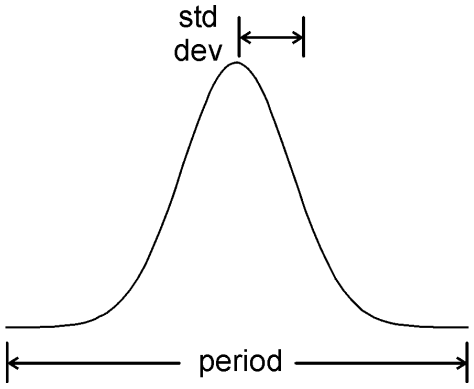


Figure 2.14: Gaussian Wave

Lorentz Pulse

A *Lorentz Pulse* function is geometric with variable half width. See figure 2.15. A *Lorentz Pulse* function is expressed by the following equation, where σ is the half width:

$$Lorentz(t) = 1/[1+(t/\sigma)^2]$$

Lorentz pulses are defined by amplitude, offset, frequency, period and half width.

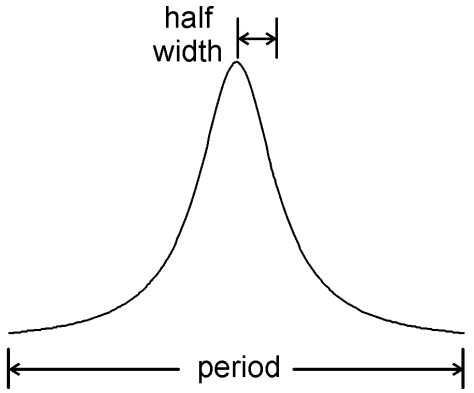


Figure 2.15: Lorentz Wave

Periodic Random Noise

A *Periodic Random Noise* function is a randomly varying signal with respect to time that repeats periodically. Random noise waveforms are defined by amplitude, offset and frequency/period. The period is the time before the noise will repeat.

AM

An *AM* function is an amplitude modulated waveform. The M-Class AWGs use an internal modulation source only. The *AM* function is defined by its modulation frequency, modulation depth percentage, and shape of the modulating waveform. The *AM* function repeats at one cycle of the modulation frequency, which equates to an integer number of cycles in the center frequency. Note that the center frequency must be an integer multiple of the modulation frequency to ensure that there will not be a discontinuity at the completion of each cycle. AM waveforms are defined by amplitude, offset, frequency, period, modulation frequency, depth and modulation shape. Note that in amplitude modulated waveforms the standard frequency attribute defines the center frequency. See figure 2.16.

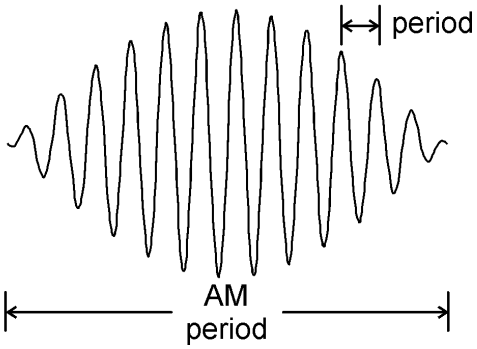


Figure 2.16: AM Wave

FM

An *FM* function is a frequency modulated waveform. The M-Class AWGs use an internal modulation source only. The *FM* function is defined by the modulation frequency, the peak frequency deviation, and the shape of the modulating waveform. The *FM* function repeats at one cycle of the modulation frequency, which equates to an integer number of cycles in the center frequency. Note that the center frequency must be an integer multiple of the modulation frequency to ensure that there will not be a discontinuity at the completion of each cycle. FM waveforms are defined by amplitude, offset, frequency, period, modulation frequency, deviation frequency and modulation shape. Note that in frequency modulated waveforms the standard frequency attribute defines the carrier frequency. See figure 2.17.

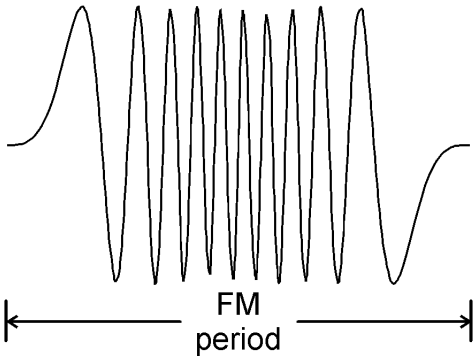


Figure 2.17: FM Wave

Multi-Tone

A *Multi-Tone* function is a combination of multiple sinusoidal tones in a single waveform. The *Multi-Tone* function is defined by a number of sinusoidal frequency tones of equal magnitude. The frequencies of up to 16 tones are user programmable. The *Multi-Tone* function repeats at the tone period which equates to the reciprocal of the minimum tone separation frequency. Note that all tones should be an integer multiple of the minimum tone separation frequency to ensure that there will not be a discontinuity at the completion of each cycle. Multi-Tone waveforms are defined by amplitude, offset, and up to 16 tone frequencies. See figure 2.18.

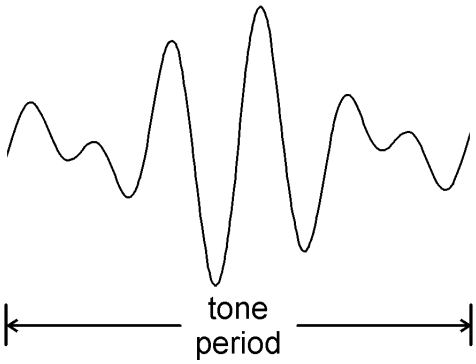


Figure 2.18: Multi-Tone Wave (3 tones)

Serial Data

A *Serial Data* function is a binary square waveform that sends a bit pattern encoded in data word format. The bit rate frequency is user programmable. The function repeats in memory at a period equal to the bit period multiplied by the number of bits in the word. The word length in bits is programmable between 4 and 64 bits. Serial data waveforms are defined by amplitude, offset, bit period, word and word length. See figure 2.19.

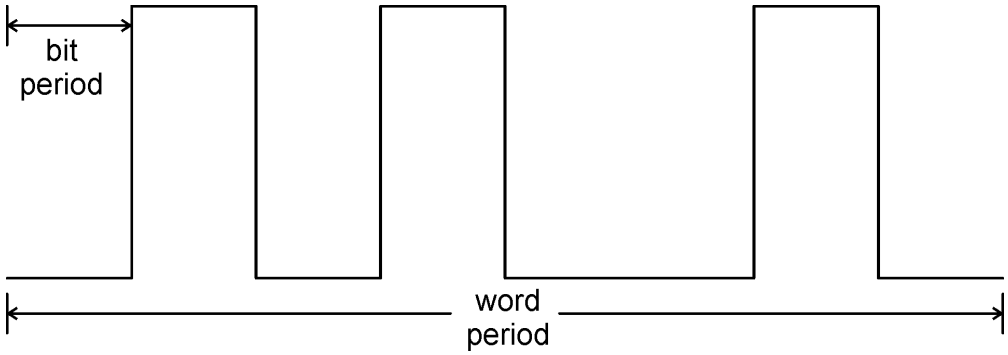


Figure 2.19: Serial Data Wave (8 bits, 01010010_2 word)

External Input

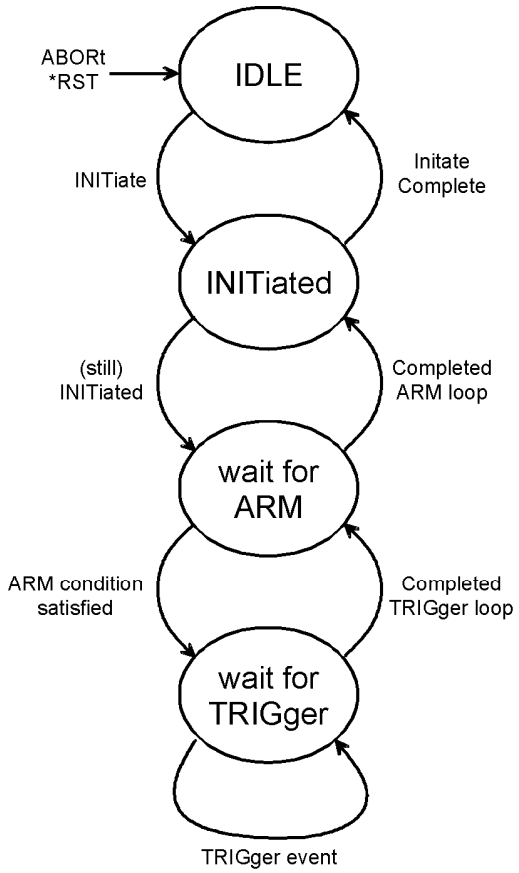
The instrument provides a multi-function front-panel input signal, EXT IN. The input impedance is user selectable to either 50Ω or 1MΩ. The input threshold level is programmable between +2V and -2V. The external input can be used for the following functions:

- Arm source
- Burst Trigger source
- Reference source for 10 MHz timebase
- Binary modulation source

Trigger and Arm

Trigger Initiate Model

The instrument uses an arm-trigger model to control signal generation. All generation cycles are started using the *Initiate Command*. In Burst Operation Mode, the instrument uses arm and trigger inputs to synchronize the signal generation. Upon receiving an “initiate”, the instrument will sequence into the “wait for arm” state. When the arm source goes active or if the arm source is set to immediate, the instrument will sequence into the “wait for trigger” state. When a trigger event is detected, the instrument will begin generating a waveform. When the waveform generation has completed, the instrument will sequence back to the idle state. An *Abort Command* or *Reset Command* will immediately stop the generation sequence and return the instrument to the idle state from any other state. The following figure shows a diagram of the trigger initiate model. It shows the arm source, trigger source, and initiate controls. See figure 2.20.



Non-Triggered:
ARM:SOURce IMMediate
TRIGger:SOURce IMMediate
INITiate starts output

Triggered:
ARM:SOURce IMMediate
TRIGger:SOURce <trigger source>
INITiate enables output
trigger event starts output

Qualified-Trigger:
ARM:SOURce <arm source>
TRIGger:SOURce <trigger source>
INITiate enables output
arm qualifies trigger event
trigger event starts output

Figure 2.20: Trigger Initiate Model

Trigger Conditions

In Burst Operation Mode, there are two trigger conditions: triggered and qualified trigger. Figure 2.21 shows the timing of both trigger conditions. In the triggered condition, the signal generation is started by a trigger event. In the qualified trigger condition, a valid trigger event must be qualified by an associated active arm condition.

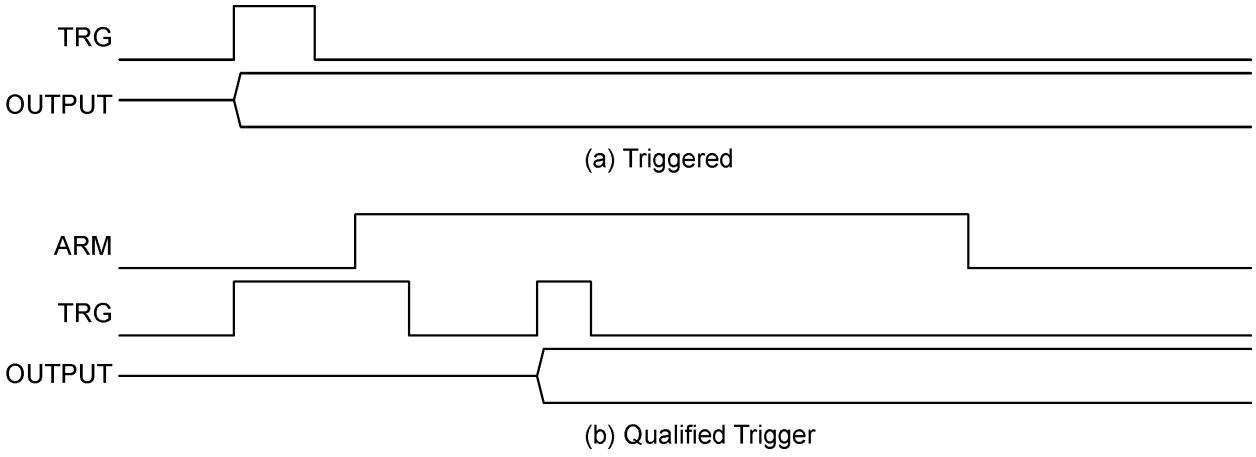


Figure 2.21: Trigger Methods

Trigger Processing

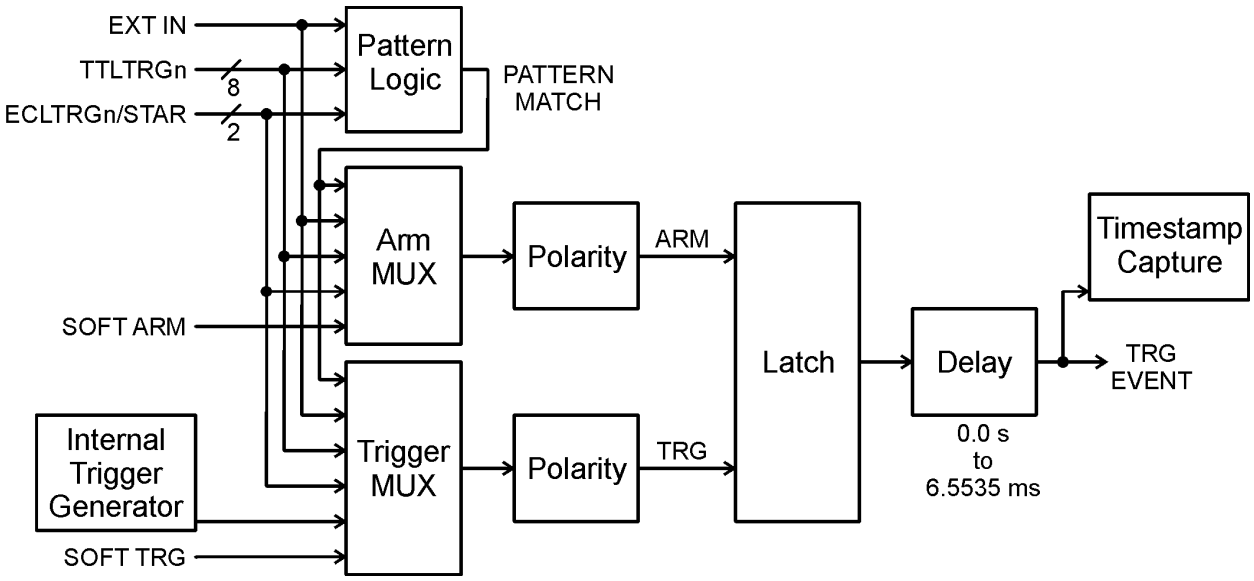


Figure 2.22: Trigger Processing

Figure 2.22 shows a diagram of the M-Class Arbitrary Waveform Generator trigger processing when operating in Burst Mode. All trigger events are edge sensitive and occur when a signal crosses a specified trigger threshold. The slope is selectable as either positive (on the rising edge) or negative (on the falling edge) for the trigger. Edge triggering is possible on the following sources:

- External Input (SMB or BNC)
- Bus Trigger 0-7 (TTL or LVDS)
- Star Trigger (PXI/PCI)
- ECL Trigger 0-1 (VXI)
- Pattern
- Internal Trigger
- Software

Pattern

The pattern match can be used as a trigger or arm source. A pattern match occurs when a set pattern is matched TRUE or FALSE. Sources for the pattern are External Input, Bus Triggers 0-7, and ECL Triggers 0-1(VXI) or Star Trigger (PXI/PCI). The states for a pattern match are HIGH or LOW. Each pattern source can also be disabled so that it will not affect the pattern matching. Trigger polarity affects pattern match (positive polarity) or pattern not match (negative polarity). For example, a pattern trigger could be set up to trigger only when TTL1 is LOW, TTL2 is HIGH, ECLT0 is LOW, and the External Input is HIGH.

Internal Trigger

Internal trigger provides a programmable pulse rate that can be used as a trigger source to generate burst signals at a specified rate. The internal trigger period is programmable between 10 µs and 100 seconds with 100 ns resolution.

Software Trigger

Software triggers occur when a software command is used to force a trigger event regardless of the selected trigger source or polarity. If manual (software) trigger source is selected, the software trigger must be used to cause a trigger event with the *Trigger Immediate* command. The *Trigger Immediate* command will immediately start generation for both triggered and qualified triggered conditions.

Trigger Timestamp

The trigger timestamp captures the time of the trigger event with 100 ns resolution within the one second on-instrument timekeeping period. With timestamps, it is possible to time or correlate multiple trigger events that caused the waveform generation.

Arm

In the qualified trigger condition, each trigger must be qualified by an associated active arm state. The arm polarity can be positive or negative. Arm sources include the following:

- External Input (SMB or BNC)
- Bus Trigger 0-7 (TTL or LVDS)
- Star Trigger (PXI/PCI)
- ECL Trigger 0-1 (VXI)
- Pattern
- Software
- Immediate (No arm qualification)

Output Channels

Output Channel Enable

Each output channel on the ZT5210 can be disabled individually. When disabled, an output is driven to the user-defined DC offset voltage for that channel. See the Output DC Offset section below for more details. See figure 2.23.

CAUTION

When disabled, the output driver is NOT tri-stated or disconnected. Instead, the output is driven to the user-programmable DC offset voltage with 50 Ω output impedance.

Output Signal Conditioning

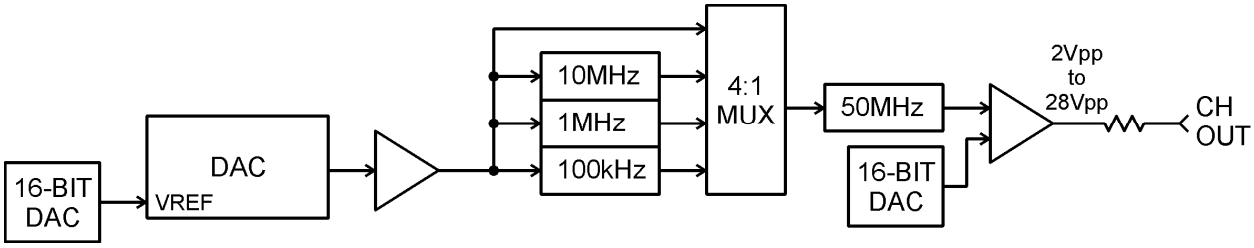


Figure 2.23: ZT5210 Output Signal Conditioning

The instrument provides signal conditioning to optimize output signal integrity. The typical analog bandwidth for the ZT5210 is DC to 50 MHz. User-configurable analog signal conditioning allows selection of output amplitude, offset and filtering.

Output Impedance

Output impedance is 50 Ω. When driving a load terminated into 50 Ω, the output signal is attenuated by 6 dB to one-half of the signal amplitude due to the loss in the output series termination resistor as shown in figure 2.24. The output voltage amplitude is programmed as peak Volts (Vp) when driving an open circuit, and as peak-to-peak Voltage (Vpp) when driving a 50 Ω load (see the *Voltage Amplitude Command*).

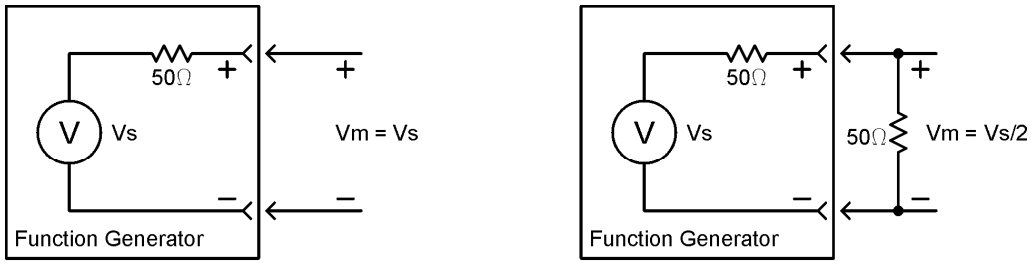
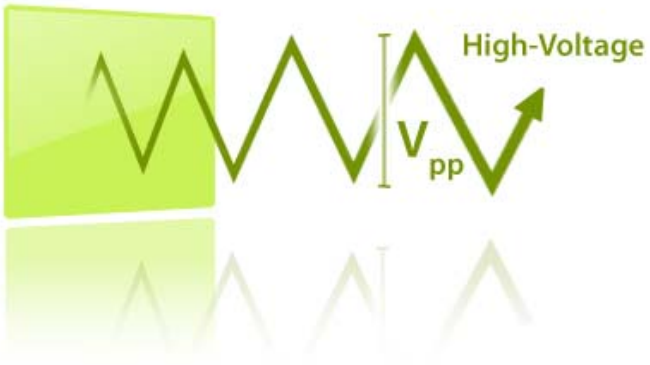


Figure 2.24: Attenuation due to Output Impedance

Output Amplitude

An adjustable DAC reference voltage allows the full-scale amplitude for the DAC to be adjusted without altering the DAC 14-bit data codes. The output voltage amplitude is programmed as peak-to-peak Volts (Vpp) (see the *Source Voltage Amplitude* command). The output voltage amplitude is adjustable from 1 Vpp to 28 Vpp (equivalent to 0.5 Vpp to 14 Vpp when driving a 50 Ω load).



Output DC Offset

The Output DC offset allows the full-scale DAC output signal to be offset (positively or negatively) with a DC voltage. The *Voltage Offset Command* is used to set the specified output channel voltage offset. The maximum output voltage, which is the sum of the output DC offset and the peak signal amplitude, is limited by the following equations:

$$|\text{offset} + \text{peak amplitude}| < \text{Vmax}$$
$$\text{Vmax} = 14\ \text{V}$$

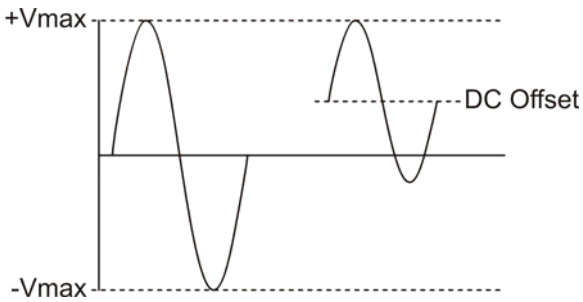


Figure 2.25: Output Voltage Maximum

Output Filter

The instrument provides four output lowpass filter options for analog signal reconstruction that are user selectable. The filters are implemented as 5-pole Bessel filters with 3 dB cutoff frequencies of 50 MHz, 10 MHz, 1 MHz and 100 kHz. A filter should be selected with a cutoff frequency below the DAC clock frequency to reject the high-frequency images within the signal.

Sync & Trigger Outputs

Sync Pulse Generation

The instrument provides two (ZT5211) or four (ZT5212) synchronization pulses (SYNC1-4) that can be routed to the front panel and backplane outputs. These digital SYNC outputs provide programmable binary waveforms that toggle at user-defined time positions in the waveform cycle. The SYNC outputs use the same memory address controller as the DAC data stream, and consequently, are tied to the same repetition period. The SYNC waveforms repeat at the same rate as one cycle of a standard function, arbitrary waveform, or waveform sequence. For each output channel, its SYNC waveform position settings must be defined before specifying the standard function, uploading the arbitrary waveform, or generating the waveform sequence.

External Output

The instrument provides a multi-function front-panel output signal source, EXT OUT. The external output is a TTL-level signal with either positive or negative polarity. The external output can be selected from the following sources:

- SYNC1-4 synchronization pulses
- Arm event
- Trigger event
- Generation complete event in Burst mode
- Operation Complete event
- Master Summary Status event
- Constant signal level
- Reference oscillator output for the selected 10 MHz timebase
- Programmable Pulse with a 16.667 ns width and repetition interval of 26.667 ns to 100 seconds
- Programmable Clock with 50% duty cycle and repetition interval of 26.667 ns to 100 seconds
- The selected common DAC sampling clock/2

Event Outputs

The instrument can drive signals over any combination of the backplane or bus trigger outputs: TTLT0–7 and ECLT0–1(VXI). Each output can be independently configured with unique source, enable and polarity controls.

Trigger Output sources include the following:

- SYNC1-4 synchronization pulses
- Arm event
- Trigger event
- Generation complete event in Burst mode
- Operation Complete event
- Master Summary Status event
- Constant signal level

Utilities and Status Reporting

Reset

Use the *Reset Command* to perform a hard reset of the instrument. This stops all waveform generation and configures the unit to its default state. See *Appendix 2, Default Reset Conditions,* for a listing of all default conditions.

Undo

The instrument can undo a reset or recall state operation. Undo returns the instrument to its state just prior to issuing a *Reset Command* or *Recall Instrument State* Command. Undo restores the output vertical settings, waveform clock and size settings, trigger settings, and generator mode settings.

Save and Recall States

The instrument can save and recall up to 14 instrument configuration states. These states record the output vertical settings, waveform clock and size settings, trigger settings, and generator mode settings. The current instrument state can be saved and recalled later. The state can either be saved on the instrument or as a file on the host processor. When stored on the instrument, states are stored in non-volatile Flash memory and are maintained when the unit is powered off. The *Reset Command*, *Save Instrument State Command*, *Recall Instrument State Command* and *Undo Command* control the instrument state configuration.

The instrument can also be configured to automatically restore any of the 14 configuration states or the reset conditions on power-up. The *System Restore Command* can be used to control the instrument power-up configuration.

Error

The instrument allows the user to see any and all system errors. The *System Error All Query* returns all 32 entries in the error log and clears the error log. Multiple errors are stored sequentially in the error log with the oldest error first. A zero value is returned for all non-error entries when there are less than 32 errors stored in the error log. The *System Error Count Query* returns the number of errors in the error log. The *System Error Next Query* returns and clears the first entry in the error log. The *System Error Report Query* returns the command string which caused the error. See *Appendix 3, System Error Codes*, for a list of error codes.

Status

The status register structure provides a common way to perform status reporting according to the IEEE 488.2 specification. This status register structure allows the user to examine the conditions of the following subsystems on the instrument: Voltage, Frequency, Self-Test, Questionable, Operation, Standard Event, and Arbitrary Waveform Generators 1 and 2.

Each status data register set contains a condition register, an event register and an event enable register. The summary output of a status data register set may be used to propagate the

status summary to the next status level and ultimately to the Status Byte. Figure 2.26 shows the complete status register structure for the ZT5210.

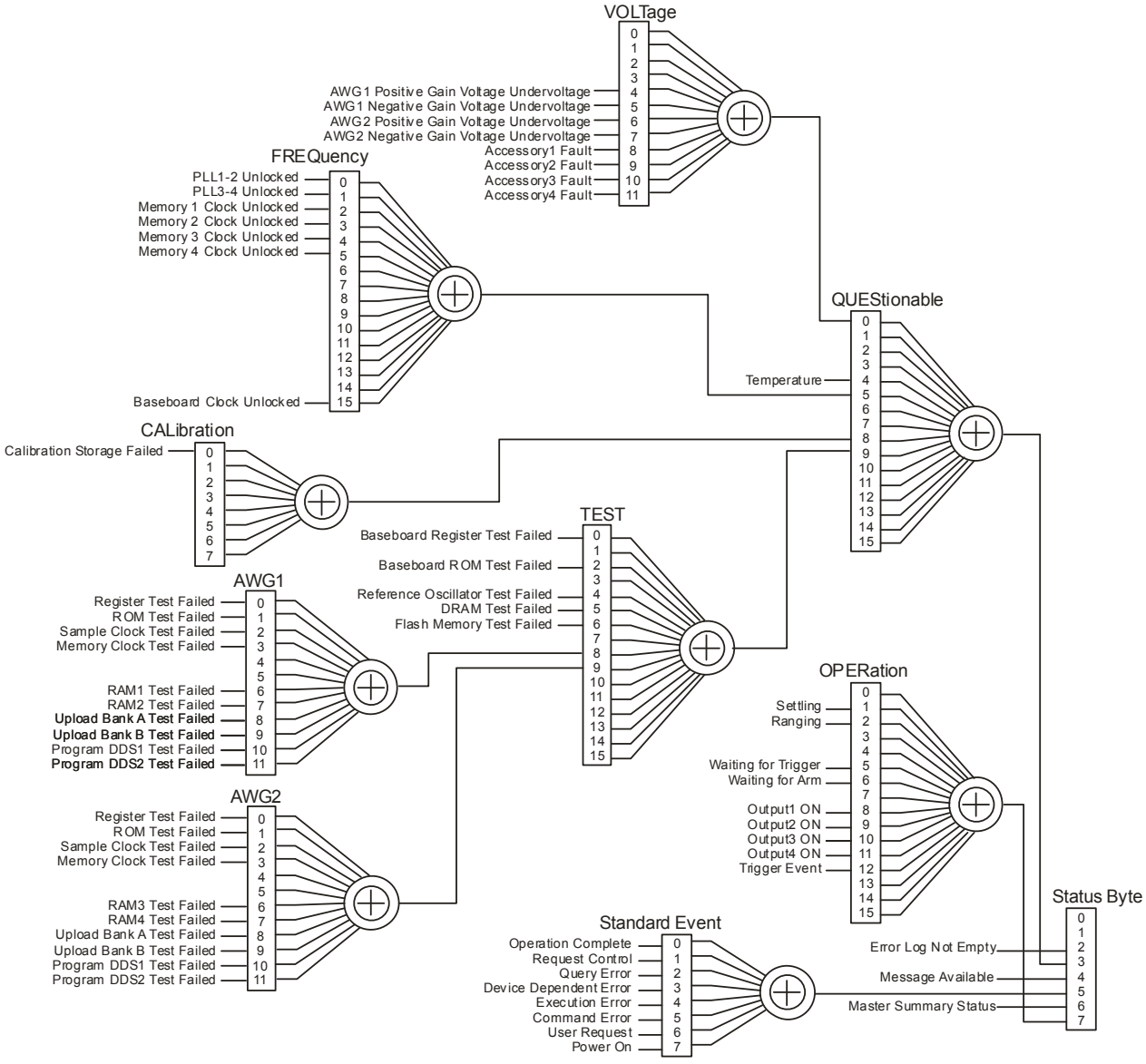


Figure 2.26: Status Register

Each individual status data register set contains the following registers with the following functionality:

Condition

A condition register provides the current device condition or state. The condition register reflects the TRUE or FALSE states in its condition bits, may range in length from 1 to 16 bits, and may contain unused bits. Unused bits will return a zero (0) value when read.

Note: Reading a condition register does not change its contents.

Event

An event register captures changes in the associated condition register. Each event bit in an event register corresponds to a condition bit in a condition register. Event registers range in length from 1 to 16 bits and may contain unused bits. Unused bits will return a zero (0) when the register is read.

An event becomes TRUE when the associated device condition transitions to a TRUE state. The event register guarantees that the application cannot miss a condition that is removed before the condition register can be read. An event register bit will be set TRUE when an associated event occurs. These bits, once set, cannot be cleared even if they do not reflect the current status of a related condition, until the event register is read by the application. Also, the instrument provides a command to clear all event registers.

Note: Event bits are cleared when read.

Enable

An enable register selects which event bits in the corresponding event register will cause a TRUE summary output when an event occurs. The summary output enabled by the event enable register is used to propagate the status summary to the next status level. Each event bit in the event register has a corresponding enable bit in the event enable register. When an event enable bit is TRUE, the corresponding event will propagate to the status summary output. Any unused bits in the event enable register correspond with unused bits in the event register. The value of unused bits is zero (0) when the event enable register is read and is ignored when written to by commands.

Self Test

The instrument can initiate an instrument self test and return any test error results as a 16-bit code (See *Test Query*). The self test is initiated on instrument power up and returns errors as shown in Table 2.2.

Hex Number Code	Error Type
0001_{16}	Baseboard Register Test Failed Bit
0002_{16}	Unused Bit
0004_{16}	ROM Test Failed Bit
0008_{16}	Unused Bit
0010_{16}	10 MHz Reference Test Failed Bit
0020_{16}	DRAM Test Failed
0040_{16}	Flash Memory Test Failed Bit
0080_{16}	Unused Bits
0100_{16}	AWG Submodule 1 Test Status
0200_{16}	AWG Submodule 2 Test Status
0400_{16}	Unused Bit
0800_{16}	Unused Bit
1000_{16}	Unused Bit
2000_{16}	Unused Bit
4000_{16}	Unused Bit
8000_{16}	Unused Bit

Table 2.2: Self-Test Errors

Calibrate

Instrument Calibration

Instructions for performing the full instrument calibration are currently available for the ZT5210 series. See the instrument calibration manual for a complete list of available commands and instructions on how to properly calibrate the instrument.

Identification and Version

Use the *ID Query* to return the instrument identification including manufacturer, model number, serial number, and firmware version. The results are returned as a block of ASCII string data up to 44 characters in length as show in the following example.

ZTEC,ZT5211PXI,100,1.00

Interface

PCI/PXI Interface

PXI Interoperability for the ZT5211PXI

The PXI module is compliant with the Peripheral Component Interconnect (PCI) Version 2.2 standard. This PCI Bus system is 33 MHz, 32-bit. The module supports universal voltage requirements of +3.3 volts and +5 volts.

The PXI module may be used in both CompactPCI (cPCI) and PCI eXtensions for Instrumentation (PXI) applications. The PXI standard is a derivative of cPCI and maintains a high level of interoperability; as a result, PXI cards may safely be used in cPCI mainframes for most applications. Compatibility issues arise because the PXI standard reserves several normally undedicated pins on the cPCI P2 connector for a selection of trigger, clock, and inter-card communication functions. Before installing the PXI module into a cPCI mainframe that uses the P2 connector, the user must ensure that pin use conflicts will not occur.

PCI Interoperability for the ZT5211PCI

The PCI module is compliant with the Peripheral Component Interconnect (PCI) Version 2.2 standard. This PCI Bus system is 33 MHz, 32-bit. The Universal PCI connector is compatible with universal +3.3V and +5V connections.

PCI Timing Expansion Connector

The PCI Timing Expansion Connector enables the synchronization of the trigger, arm and clock timebase for multiple PCI modules from ZTEC® Instruments. When using this configuration, one board acts as a timing and arm source; all of the other boards are receivers. The Trigger0-7 inputs for the PCI module mating connector are active low, with internal pull-up resistors. The outputs are open-collector, and consequently can be driven by any module. In most cases, the one source will drive all others in the system, but a wired-or configuration can also be used.

When installing multiple PCI modules, use the ribbon cable (FFSD-10-D-6.00-01-N-D4) to connect to the PCI Timing Expansion Connectors on the top of the modules (See Figure 3.1). The ribbon cable can be matched in length to achieve a zero clock skew between modules. Table 3.1 shows the pin out for the mating connectors to the PCI Timing Expansion Connector.

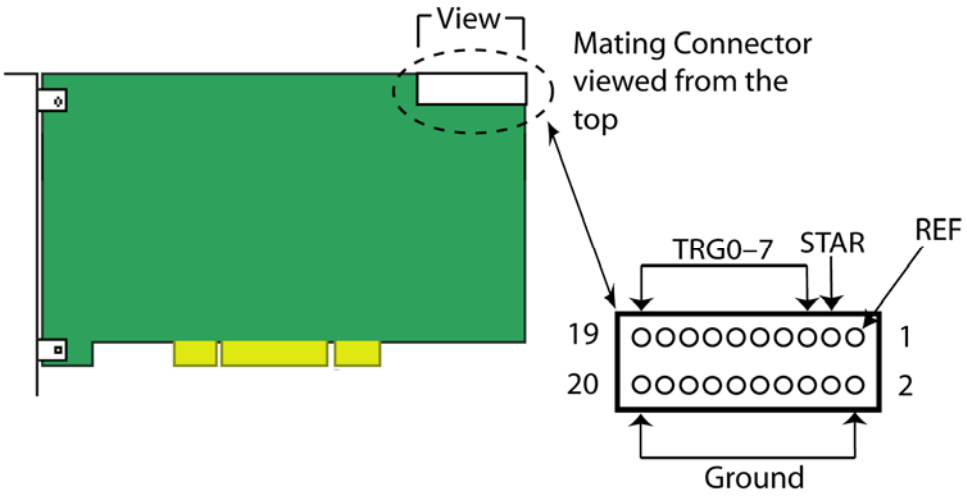


Figure 3.1: PCI Timing Expansion Connector

Pin	Function	Pin	Function
1	REF	**2**	GND
3	STAR	**4**	GND
5	TRG7	**6**	GND
7	TRG6	**8**	GND
9	TRG5	**10**	GND
11	TRG4	**12**	GND
13	TRG3	**14**	GND
15	TRG2	**16**	GND
17	TRG1	**18**	GND
19	TRG0	**20**	GND

Table 3.1: PCI Timing Expansion Connector Pin Out

VXI Interface

Interface Description

The ZT5210VXI is a message-based VXIbus module that supports both VXIbus Instrument and VXIbus 488.2 Instrument protocols. It is compliant with the instrument specifications outlined in the VXI-1 Revision 1.4 and IEEE Standard 488.2-1992 specifications. In accordance with these specifications, the VXI module supports a number of levels of communication protocols including low-level VXIbus word-serial commands, IEEE 488.2 common commands.

LXI Interface

Interface Description

ZTEC® LXI devices are packet-based ethernet devices that are compliant with the IEEE 802.3 Ethernet standard. The ZT5210LXI is compliant with the LXI Functional Class C standard, Revision 1.2, as defined by the LXI consortium: http://www.lxistandard.org/.

webLXI

LXI device information can be read and changed through the webLXI interface hosted on the LXI instrument. The website can be viewed by going to the instrument's IP address in any web browser (http://[IPADDRESS]). Settings that can be changed are password protected; default is no password and user name is "webLXI". The password can be reset to default using the reset button on the back of the instrument.

Some fields in the webLXI interface start with blank entries. The functionality such fields represent is disabled. Entering a value here will enable the functionality. Settings not relevant to the device's IP settings which ask for an IP address can later be disabled by entering '0.0.0.0' in the webLXI field or by pressing the reset button at the back of the instrument.

IP Address

ZTEC® LXI devices are shipped with LXI standard IP options. The IP configuration can be changed using the webLXI interface and is capable of two states. By default, the device is set to the Auto-TCP/IP state, which uses DHCP or Auto-IP if no DHCP server is found within 30 seconds. Auto-IP will provide an IP from the range 169.254.[1-254].[1-254] for private networking. Polling for a new DHCP server occurs every 2 minutes in the Auto-TCP/IP state. The alternative state is Manual, in which the IP configuration is set to a user-appointed static IP.

Note: The static IP 169.254.0.100 is reserved and must be unallocated for successful instrument startup.

In both states, ZTEC® LXI devices are capable of detecting duplicate IP addresses. If there is a duplicate IP found while in the Manual state, the device cannot be used until reset or until the duplicate IP becomes available. IP settings can be reset to default using the reset button on the back of the instrument.

The IP address can be discovered using ZFind™ or any other LXI compliant discovery utility. Additionally, during startup, the instrument configuration may be viewed by attaching a monitor to the rear VGA port.

Ports

ZTEC® LXI instruments use several IP ports to communicate to the host computer. These ports must be open on the host computer and network or the instrument will not be properly accessible.

Port #	Use
3030	Driver Interface; all drivers, GUI
3333	VXI-11
5064	EPICS Server Port; this port may be changed through the webLXI interface
5065	EPICS Repeater Port; this port may be changed through the webLXI interface
8080	webLXI

Table 3.2: LXI Ports

Sockets

As shown in the Ports section, most communication to the instrument is done using the 3030 port. Currently, this port supports only a single socket connection. A socket is created any time a handle is created to the instrument, and is closed when the handle is closed.

Sockets to this port have a timeout; after 60 seconds with no communication, the socket will close itself. This communication is done as a background process and does not require user code. This is done so that if there are network problems, or other issues which result in an improperly closed socket, the instrument will allow new connections to be created. The LXI socket timeout can be extended to 25 minutes by calling zbind_tmo_disable(). zbind_tmo_enable() will reset the timeout to the default 60 seconds. Extending the timeout may be desirable to prevent timeouts from code that is paused for debugging purposes.

NTP Server Address

A Network Time Protocol (NTP) Server may be provided in order to keep the instrument set to the current time. A link to the National Institute of Standards and Technology's (NIST) list of NTP Servers is provided on the webLXI interface. Once connected, an NTP Server will be polled every ten minutes to adjust the time and also upon device initialization.

Hardware Triggering

ZTEC® LXI instruments implement hardware triggering; this enables the synchronization of trigger, arm, and clock timebase for multiple devices. The wired trigger bus (WTB) hardware generally conforms to the standard defined by LXI standard version 1.2 section 5.

The WTB can be operated in either Driven or Wired-Or mode. Use the *Output LXI Mode Command* to configure the mode of operation.

Driven Mode

This mode of operation provides point-to-multipoint operation of the bus. One device initiates a trigger event to one or more receiving devices.

Wired-Or Mode

This mode of operation provides multipoint-to-multipoint operation of the bus. In this mode, one device is configured as the Wired-Or Bias device for the trigger channel. Other devices participating in the wired trigger require two driver devices per channel in order to change the state of the channel.

Connections

When connecting the hardware trigger bus use appropriate cables and terminators. Visit the LXI consortium for a list of part vendors. Devices can be connected in daisy-chain, star, or hybrid star configurations as defined by the LXI standard. Table 3.3 below shows the pin out for the WTB connectors on the back panel of the ZTEC® LXI device.

Pin	Signal	Pin	Signal
1	+3.3V	14	LXI0p
2	+3.3V_Return	15	LXI0n
3	LXI1p	16	Reserved
4	LXI1n	17	LXI2p
5	GND	18	LXI2n
6	LXI3p	19	GND
7	LXI3n	20	LXI4p
8	GND	21	LXI4n
9	LXI5p	22	GND
10	LXI5n	23	LXI6p
11	Reserved	24	LXI6n
12	LXI7p	25	Reserved
13	LXI7n	Connector Shell	Chassis

Table 3.3: LXI WTB Connector Pin Out

Software

There are multiple types of software interfaces available for use with the ZT5210. Direct SCPI calls can be made using the zbind hardware binding layer, through visa calls (VXI and Visa PCI/PXI), using VXI-11 (LXI) or through the ZFind™ application (see the ZFind User's Guide). There are also ANSI-C based drivers available that provide a more user-friendly and intuitive interface to instrument functionality. The drivers provide grouped functions (using multiple SCPI commands), simple waveform upload and download functionality, and error checking. Additionally, driver wrappers are provided in multiple interfaces such as IVI, LabVIEW and COM.

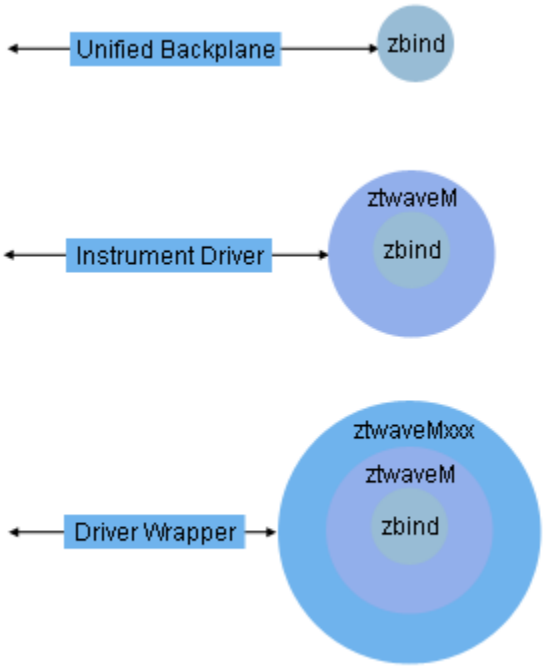


Figure 3.2: Driver Interface Layers

All instrument software communication layers are built on top of each other to ensure identical functionality (see Figure 3.2). The zbind unified backplane layer provides a level of abstraction from the hardware so that PCI, PXI, VXI and LXI instruments all share the same interface. The instrument driver layer encapsulates this abstraction and creates a common set of functions for all M-Class AWGs. Driver wrappers provide identical functionality to the instrument drivers, but allow the drivers to be accessed through alternate interfaces. Specific driver command information is available in the M-Class AWG Driver Manual.

A Graphical User Interface (GUI) is also provided for ease-of-use and quick instrument set up. Additional ZWave® information is available in the ZWave® M-Class User's Guide.

Command Reference

This chapter describes IEEE 488.2 Common (*) commands and Standard Commands for Programmable Instruments (SCPI) applicable to the instrument.

Common Command Format

The IEEE 488.2 standard defines the Common Commands that perform functions like reset, self-test, status byte query, etc. Common commands are four or five characters in length, always begin with the asterisk character (*), and may include one or more parameters. The command keyword is separated from the first parameter by a space character. Some examples of Common Commands are shown below:

- *RST
- *CLS
- *STB?

SCPI Command Format

Standard Commands for Programmable Instruments (SCPI) perform functions like setting parameters, performing measurements, querying instrument states, and retrieving data. A subsystem command structure is a hierarchical structure that usually consists of a top level (or root) command, one or more lower-level subcommands, and their parameters. The following example shows part of a typical subsystem:

[SOURce<number>:]
WAVeform
:SWITch:MODE SEAMless | INSTantaneous
:POINts?

[SOURce:] is the root command with <number> as a parameter, :WAVeform is the second-level subcommand, and :SWITch and :POINts? are third-level commands/queries. :SWITch:MODE has SEAMless and INSTantaneous as a parameter.

Command Separator

A colon (:) always separates one command from the next lower level command as shown below:

[SOURce<n>:]BURSt:COUNt?

Colons separate the root command from the second-level command ([SOURce<n>:]BURSt), and the second-level from the third-level query (BURSt:COUNt?).

Abbreviated Commands

The command syntax shows most commands as a mix of upper and lower case letters. The upper case letters indicate the abbreviated spelling for the command. For shorter program lines, send only the abbreviated form. For better program readability, you may send the entire command. The instrument will only accept either the abbreviated form or the entire command.

For example, if the command syntax shows *IMPedance?*, then *IMP?* and *IMPedance?* are both acceptable forms. Other forms of *IMPedance?*, such as *IM?*, will generate an error. The commands are not case sensitive. Therefore, *IMPEDANCE?* and *ImPeDaNcE?* are acceptable.

Implied Commands

Implied commands are those which appear in square brackets ([ ]) in the command syntax. (Note that the brackets are not part of the command and are not sent to the instrument.) If a root level and second-level command are sent, but not a third-level implied command, the instrument assumes use of an implied command. Examine the portion of the [SOURce:] subsystem shown below:

[SOURce<n>:]
WAVeform
:POINts?

The first-level command [SOURce<n>:] is an implied command. To query the instrument's channel 1 waveform size, send either of the following command statements:

SOUR1:WAV:POIN? *or* **WAV:POIN?**

Parameters

The following table contains explanations and examples of parameter types.

Parameter Type	Explanations and Examples
Mask	Bit mask (<mask>) where every bit represents a state.
Numeric	Accepts all commonly used decimal representations of numbers including optional signs, decimal points, and scientific notation. 123 or 1.23E2; -123 or -1.23E2; .123, 1.23E-1, or 1.23000E-01. Accepts all commonly used suffixes with decimal representations of numbers, including optional signs and decimal points. .123S or 123MS; 1234OHM or 1.234KOHM. Voltage = "UV" for E-6, "MV" for E-3, "V"for E0, "KV" for E3 Percent = "PCT" Ohms = "OHM", "KOHM" for E3, "MOHM" for E6. Frequency = "HZ" for E0, "KHZ" for E3, "MHZ" for E6, "GHZ" for E9. Time = "PS" for E-12, "NS" for E-9, "US" for E-6, "MS" for E-3, "S" for E0. Phase = “DEG” for (π/180) degrees or “RAD” for radians Special cases include MIN and MAX. MIN (selects minimum value available), and MAX (selects maximum value available). Numeric values are either Integers or Floats. **Note:** All command parameters represented as floating point numbers sent to the ZT5210 must have 12 digits or less, including before and after the decimal point. Examples: 123456.789012 or 0.12345678901
Discrete	Selects from a finite number of values. These parameters use mnemonics to represent each valid setting. An example is the [SENSe:]INPut<n>:COUPling <mode> command, where <mode> can be AC or DC. Another example is a single binary condition that is either true or false. 1 or ON; 0 or OFF

Table 4.1: Parameter Types

Optional Parameters

Parameters shown within square brackets ([ ]) are optional parameters. (Note that the brackets are not part of the command and are not sent to the instrument.) If you do not specify a value for an optional parameter, the instrument chooses a default value.

Parameters Out of Range

An out of range parameter is automatically adjusted to the closest acceptable value. For example, if **BURSt:COUNt 0** is entered, the value is set to 1 (lowest available setting).

Linking Commands

Linking IEEE 488.2 Common Commands with SCPI Commands

Use a semicolon between the commands.

For example: ***CLS;*RST;*TRG**

Linking Multiple SCPI Commands

Use a semicolon and a colon between the commands.

For example: **SOUR1:FUNC SINE;:SYST:ERR?**

SCPI also allows several commands within the same subsystem to be linked with a semicolon. For example: **SOUR1:FUNC SINE;:SOUR1:VOLT 10** *or* **SOUR1:FUNC SINE;VOLT 10**

IEEE 488.2 Common Commands

The following is an alphabetic list of IEEE 488.2 Common Commands.

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Clear Status Command
*CLS</td><td>The clear status command clears all event registers, the request for OPC flag, and all status queues (except the response queue).

Command Syntax
*CLS

Query Syntax
None

Parameters
None</td></tr>
<tr><td>Event Status Enable Command
*ESE

Event Status Enable Query
*ESE?</td><td>Sets and returns the state of the event status enable register. The event status enable register allows the enabled standard events to affect the event summary status bit within the status byte.

Command Syntax
*ESE <enable>

Query Syntax
*ESE? → <enable>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><enable></td><td>8-bit mask</td><td>0 to 255
Bit 0: Operation Complete
Bit 1: Request Control
Bit 2: Query Error
Bit 3: Device Dependent Error
Bit 4: Execution Error
Bit 5: Command Error
Bit 6: User Request
Bit 7: Power on</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Event Status Register Query
*ESR?</td><td>The event status register query returns the state of the event status register. The event status register provides the standard event status information.

Command Syntax
None

Query Syntax
*ESR? → <state>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><state></td><td>8-bit mask</td><td>0 to 255
Bit 0: Operation Complete
Bit 1: Request Control
Bit 2: Query Error
Bit 3: Device Dependent Error
Bit 4: Execution Error
Bit 5: Command Error
Bit 6: User Request
Bit 7: Power on</td></tr></table></td></tr>
<tr><td>Identification Query
*IDN?</td><td>The identification query returns the instrument identification information. The response contains four fields separated by commas in the form:

“Manufacturer, model number, serial number, firmware revision level.” Ex: ZTEC, ZT5211PXI,100,1.00

Command Syntax
None

Response Syntax
*IDN? → <id_string>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><id_string></td><td>String</td><td>See above</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Operation Complete Command
*OPC

Operation Complete Query
*OPC?</td><td>The command sets the request for the operation complete flag when all pending operations have completed. When all operations have completed, the operation complete bit in the event status register will be set.

The query returns a 0 to indicate that all pending operations have not completed and a 1 to indicate that all pending operations have completed. This query blocks all instrument commands until complete and should be used with caution.

Command Syntax
*OPC

Query Syntax
*OPC? → <state>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><state></td><td>Discrete</td><td>0 All operations are not complete
1 All operations complete</td></tr></table></td></tr>
<tr><td>Recall Instrument State Command
*RCL</td><td>Recalls the selected saved instrument state from non-volatile memory.

Command Syntax
*RCL <number>

Query Syntax
None

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><number></td><td>Integer</td><td>1 to 14</td></tr></table></td></tr>
<tr><td>Reset Command
*RST</td><td>Performs a hardware reset that returns the instrument to the initial default condition. Status registers are not cleared.

Command Syntax
*RST

Query Syntax
None

Parameters
None</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Save Instrument State Command
*SAV</td><td>Stores the current state of the instrument to the selected storage index in non-volatile memory.

Command Syntax
*SAV <number>

Query Syntax
None

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><number></td><td>Integer</td><td>1 to 14</td></tr></table></td></tr>
<tr><td>Service Request Enable Command
*SRE

Service Request Enable Query
*SRE?</td><td>Selects or returns the enabled bits for the Status Byte. The parameter is a bit mask which enables the corresponding status byte bits.

Command Syntax
*SRE <enable>

Query Syntax
*SRE? → <enable>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><enable></td><td>8-bit mask</td><td>0 to 255
Bit 0: Unused
Bit 1: Unused
Bit 2: Error Log Not Empty
Bit 3: Questionable Summary
Bit 4: Message Available
Bit 5: Standard Event Summary
Bit 6: Master Summary Status
Bit 7: Operation Summary</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Byte Query
*STB?</td><td>Returns the Status Byte. The parameter is a bit mask which describes the state of the corresponding status byte bits.

Command Syntax
None

Query Syntax
*STB? → <mask>

Parameters
<table><tr><th>Name</th><th>Type</th><th>Range</th></tr><tr><td><mask></td><td>8-bit mask</td><td>0 to 255
Bit 0: Unused
Bit 1: Unused
Bit 2: Error Log Not Empty
Bit 3: Questionable Summary
Bit 4: Message Available
Bit 5: Standard Event Summary
Bit 6: Master Summary Status
Bit 7: Operation Summary</td></tr></table></td></tr>
<tr><td>Trigger Immediate Command
*TRG</td><td>Causes an immediate trigger event regardless of selected trigger source. This event will override both the trigger and arm events, if enabled. If enabled, the trigger outputs on the external output or backplane bus will also toggle when a trigger immediate command is issued.

Command Syntax
*TRG

Query Syntax
None

Parameters
None</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Test Query
*TST?</td><td>Initiates an instrument self test and returns the test results as a 16-bit code. The self test is initiated on instrument power up.

Command Syntax
None

Query Syntax
*TST? → <code>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><code></td><td>16-bit mask</td><td>0 to 65535
Bit 0: Baseboard Test Failed
Bit 1: Unused
Bit 2: ROM Test Failed
Bit 3: Unused
Bit 4: Reference Oscillator Test Failed
Bit 5: DRAM Test Failed
Bit 6: Flash Memory Test Failed
Bit 7: Unused
Bit 8: AWG Submodule 1 Test Failed
Bit 9: AWG Submodule 2 Test Failed
Bits 10-15: Unused</td></tr></table></td></tr>
<tr><td>Wait to Continue Command
*WAI</td><td>Allows the user to force the interface to wait until operations are complete before resuming.

Command Syntax
*WAI

Query Syntax
None

Parameters
None</td></tr>
</table>

SCPI Commands and Queries

The SCPI Commands are presented in an alphabetic list below. Each SCPI command parameter table includes parameter name, parameter type, and range of values. The parameter type follows the definition and information given in Table 4.1.

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Abort Command</td><td>Terminates waveform generation. When an abort is received the instrument will end any on-going generation activity and return to its idle state. The unit start state can be queried from the Status Register

Command Syntax
ABORt

Query Syntax
None

Parameters
None</td></tr>
<tr><td>Accessory Identify Query</td><td>Queries the identification string of the ZTEC® accessory attached to OUTPn.

Command Syntax
None

Query Syntax
[SOURce<n>:]ACCessory:IDENtify? → <id_string>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><id_string></td><td>String</td><td>N/A</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>AM Depth Command

AM Depth Query</td><td>Sets or queries the Amplitude Modulation Depth. The AM shape, frequency and depth are used to create the modulating sidebands. Carrier waves are always sine waves and are adjusted using the Period, Frequency, Voltage Offset and Voltage Amplitude commands. AM waveforms are only recalculated on a Function Shape Command.

Command Syntax
[SOURce<n>:]AM[:DEPTh] <depth>

Query Syntax
[SOURce<n>:]AM[:DEPTh]? → <depth>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><depth></td><td>Float</td><td>Fraction of total waveform amplitude
MINimum 0.0
MAXimum 1.00
May also be entered as a percentage: 0PCT to 100PCT</td></tr>
</table>
</td></tr>
<tr><td>AM Frequency Command

AM Frequency Query</td><td>Sets or queries the Amplitude Modulation Frequency. The ratio of the Carrier Frequency to the Modulation Frequency must be less than 1000:1. The AM shape, frequency and depth are used to create the modulating sidebands. Carrier waves are always sine waves and are adjusted using the Period, Frequency, Voltage Offset and Voltage Amplitude commands. AM waveforms are only recalculated on a Function Shape Command.

Command Syntax
[SOURce<n>:]AM:FREQuency <freq>

Query Syntax
[SOURce<n>:]AM:FREQuency? → <freq>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><freq></td><td>Float</td><td>Modulation Frequency in Hertz.
MINimum 1 Hz
MAXimum Smaller of 1 MHz or Center Frequency</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>AM Shape Command

AM Shape Query</td><td>Sets or queries the Amplitude Modulation Shape. The AM shape, frequency and depth are used to create the modulating sidebands. Carrier waves are always sine waves and are adjusted using the Period, Frequency, Voltage Offset and Voltage Amplitude commands. AM waveforms only recalculated on a Function Shape Command.

Command Syntax
[SOURce<n>:]AM:SHAPe <shape>

Query Syntax
[SOURce<n>:]AM:SHAPe? → <shape>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><shape></td><td>Discrete</td><td>SINE Sine
SQUare Square
TRIangle Triangle
RUP Ramp Up
RDOWn Ramp Down</td></tr>
</table>
</td></tr>
<tr><td>Arm Command

Arm Query</td><td>Arms or disarms the unit through software when manual arm source is selected. The instrument will begin trigger detection when armed. When disarmed, the unit ignores triggers. The Arm Query returns the current arm condition.

Command Syntax
ARM[:IMMediate] <state>

Query Syntax
ARM? → <state>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><state></td><td>Discrete</td><td>ON or 1 (Arm)
OFF or 0 (Disarm)</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Arm Polarity Command

Arm Polarity Query</td><td>Sets or queries the active arm state. If an arm source is selected and the state of the selected source matches the Arm Polarity state, the unit will arm. The following considerations apply when setting the arm polarity:

• POSitive state defines the active state as the selected source in its high state.
• NEGative state defines the active state as the selected source in its low state.

Command Syntax
ARM:POLarity <polarity>

Query Syntax
ARM:POLarity? → <polarity>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><polarity></td><td>Discrete</td><td>NEGative (negative polarity)
POSitive (positive polarity)</td></tr></table></td></tr>
<tr><td>Arm Source Command

Arm Source Query</td><td>Sets or queries the source that will be used to arm the instrument. For example, if the Arm Source Command is sent with EXTernal, the front panel EXT IN signal will be used to arm the unit. If an immediate output is desired regardless of the arm state, Arm Source Command can be sent as IMMediate.

Command Syntax
ARM:SOURce <source>

Query Syntax
ARM:SOURce? → <source>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><source></td><td>Discrete</td><td>TTLTrg<n> TTL trigger line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7
ECLTrg<n> VXIbus ECL trigger line, where <n> may be 0 or 1. VXI only.
MANual Manual arm
IMMediate Bypass arm detection
EXTernal Front panel EXT IN signal
Pattern Arm based on source pattern
Star Trigger PXI/PCI only.</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Binary Modulation View Command

Binary Modulation View Query</td><td>Sets the current binary modulation buffer view for waveform data operations. The view effects which buffer is used for all data transfer operations such as write, read, copy, scale, etc that are performed on channels in Binary Modulation Mode. The view does not affect the Binary Modulation State.

Command Syntax
BMODulation:VIEW <view>

Query Syntax
BMODulation:VIEW? → <view>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><view></td><td>Discrete</td><td>0 or 1</td></tr></table></td></tr>
<tr><td>Binary Modulation Source Command

Binary Modulation Source Query</td><td>Sets or queries the binary source that will be used to modulate the instrument’s output. For example, if the source is set to EXTernal, the front panel EXT IN signal will be used to modulate the output.

Command Syntax
BMODulation:SOURce <source>

Query Syntax
BMODulation:SOURce? → <source>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><source></td><td>Discrete</td><td>TTLTrg<n> TTL trigger line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7
ECLTrg<n> VXIbus ECL trigger line, where <n> may be 0 or 1. VXI only.
MANual Manual modulation
EXTernal Front panel EXT IN signal
Pattern Modulate based on source pattern
Star Trigger PXI/PCI only.</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Binary Modulation State Command

Binary Modulation State Query</td><td>Sets or queries the current binary modulation state. Set state is only valid when Binary Modulation Source is set to MANual. Binary Modulation Query returns the current modulation condition.

Command Syntax
BMODulation[:STATe] <state>

Query Syntax
BMODulation[:STATe]? → <state>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><state></td><td>Discrete</td><td>0 Low Binary State
1 High Binary State</td></tr>
</table>
</td></tr>
<tr><td>Burst Count Command

Burst Count Query</td><td>Sets or queries the Burst Count. The count selects how many cycles of a waveform are played upon a burst trigger event. This only applies to channels that are set to Burst Mode.

Command Syntax
[SOURce<n>:]BURSt:COUNt <number>

Query Syntax
[SOURce<n>:]BURSt:COUNt? → <number>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><number></td><td>Integer</td><td>Number of Cycles, 1 to 65535</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Calibration Date Query</td><td>Returns the date of the instrument’s last factory calibration.

Command Syntax
None

Query Syntax
CALibration:DATE? → <month>,<day>,<year>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><month></td><td>Integer</td><td>1-12</td></tr><tr><td><day></td><td>Integer</td><td>1-31</td></tr><tr><td><year></td><td>Integer</td><td>0-65,535c.e.</td></tr></table></td></tr>
<tr><td>Calibration Default Command</td><td>This function returns the instrument calibration data to default.

Warning!
Before executing this function be sure to read the instrument calibration manual.

Command Syntax
CALibration:DEFault <key>

Query Syntax
None

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><key></td><td colspan="2">Refer to Calibration Manual</td></tr></table></td></tr>
<tr><td>Calibration Restore Command</td><td>Restores the reserved default calibration data. This will reset all calibration data to those values saved by the Calibration Save Command.

Warning!
Before executing this function be sure to read the instrument calibration manual.

Command Syntax
CALibration:RESTore

Query Syntax
None

Parameters
None</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>DAC Clock Common Command

DAC Clock Common Query</td><td>Sets or queries the common DAC Clock source. The common source is used for all channels that are in Common DAC Clock Mode.

Command Syntax
[SOURce:]DAC:CLOCk:COMMon <source>

Query Syntax
[SOURce:]DAC:CLOCk:COMMon? → <source>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><source></td><td>Discrete</td><td>OUTP1 Output Channel 1
OUTP2 Output Channel 2
OUTP3 Output Channel 3
OUTP4 Output Channel 4</td></tr>
</table></td></tr>
<tr><td>DAC Clock Frequency Command

DAC Clock Frequency Query</td><td>Sets or queries the selected channel’s DAC Clock Frequency. Each channel has an independent DAC clock unless the DAC Clock Mode is set to Common for that channel. DAC Clock Frequency Command/Query is available regardless of the DAC Clock Mode; channels set to use the Common source will not use their DAC Clock’s frequency.

Command Syntax
[SOURce<n>:]DAC:CLOCk:FREQuency <freq>

Query Syntax
[SOURce<n>:]DAC:CLOCk:FREQuency? → <freq>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><freq></td><td>Float</td><td>Clock frequency in Samples/second.
MINimum 200 S/s
MAXimum 200 MS/s</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>DAC Clock Mode Command

DAC Clock Mode Query</td><td>Sets or queries the selected channel's DAC Clock Mode. When the clock mode is Independent, the channel's sample rate may be set independently. When the clock mode is Common, the channel's sample rate is determined by the DAC Clock Common Source.

Command Syntax
[SOURce<n>:]DAC:CLOCk:MODE <mode>

Query Syntax
[SOURce<n>:]DAC:CLOCk:MODE? → <mode>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><mode></td><td>Discrete</td><td>COMMon Common Clock
INDependent Independent</td></tr>
</table>
</td></tr>
<tr><td>Duty Cycle Command

Duty Cycle Query</td><td>Sets or queries the selected channel's positive duty cycle. Duty cycle is only applicable for square and pulse waveforms. Positive duty cycle and positive width are related by the function:

width = dcycle*period

Command Syntax
[SOURce<n>:]DCYCle <duty>

Query Syntax
[SOURce<n>:]DCYCle? → <duty>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><duty></td><td>Float</td><td>Fraction of total period.
MINimum 0.0
MAXimum 1.0
May also be entered as a percentage: 0PCT to 100PCT</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Filter Frequency Command

Filter Frequency Query</td><td>Sets or queries the selected channel's filter frequency. The channel's output is filtered using a lowpass filter of the selected frequency. See Appendix 1: Specifications for more detailed filter information.

Command Syntax
[SOURce<n>:]FILTer[:LPASs]:FREQuency <freq>

Query Syntax
[SOURce<n>:] FILTer[:LPASs]:FREQuency? → <freq>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><freq></td><td>Discrete</td><td>50e6 or MAXimum 50 MHz Filter
10e6 10 MHz Filter
1e6 1 MHz Filter
100e3 or MINimum 100 kHz Filter</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>FM Deviation Command

FM Deviation Query</td><td>Sets or queries the Frequency Modulation Deviation. The FM shape, frequency and deviation are used to create the modulating signal. Carrier waves are always sine waves and are adjusted using the Period, Frequency, Voltage Offset and Voltage Amplitude commands. FM waveforms only recalculated on a Function Shape Command.

The FM deviation frequency must be greater than the FM modulation frequency. If the FM deviation frequency is not a multiple of the FM modulation frequency it will cause discontinuities in the waveform

Command Syntax
[SOURce<n>:]FM[:DEViation] <deviation>

Query Syntax
[SOURce<n>:]FM[:DEViation]? → <deviation>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><deviation></td><td>Float</td><td>Deviation Frequency in Hertz.
MINimum 1 Hz
MAXimum Smaller of 1 MHz
or Center Frequency</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>FM Frequency Command

FM Frequency Query</td><td>
Sets or queries the FM Modulation Frequency. The ratio of the Carrier Frequency to the Modulation Frequency must be less than 1000:1. The FM shape, frequency and deviation are used to create the modulating signal. Carrier waves are always sine waves and are adjusted using the Period, Frequency, Voltage Offset and Voltage Amplitude commands. FM waveforms only recalculated on a Function Shape Command.

The FM modulation frequency must be less than the FM deviation frequency. If the FM deviation frequency is not a multiple of the FM modulation frequency it will cause discontinuities in the waveform

Command Syntax
[SOURce<n>:]FM:FREQuency <freq>

Query Syntax
[SOURce<n>:]FM:FREQuency? → <freq>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><freq></td><td>Float</td><td>Modulation Frequency in Hertz.
MINimum 1 Hz
MAXimum Smaller of 1 MHz or Center Frequency</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>FM Shape Command

FM Shape Query</td><td>Sets or queries the Frequency Modulation Shape. The FM shape, frequency and deviation are used to create the modulating signal. Carrier waves are always sine waves and are adjusted using the Period, Frequency, Voltage Offset and Voltage Amplitude commands. FM waveforms only recalculated on a Function Shape Command.

Command Syntax
[SOURce<n>:]FM:SHAPe <shape>

Query Syntax
[SOURce<n>:]FM:SHAPe? → <shape>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><shape></td><td>Discrete</td><td>SINE Sine
SQUare Square
TRIangle Triangle
RUP Ramp Up
RDOWn Ramp Down</td></tr>
</table>
</td></tr>
<tr><td>Format Byte Order Command

Format Byte Order Query</td><td>Sets or returns the current byte order setting. Normal byte order is MSB first. Swapped byte order is LSB first.

Command Syntax
FORMat:BORDer <order>

Query Syntax
FORMat:BORDer? → <order>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><order></td><td>Discrete</td><td>SWAPped Swapped byte order (LSB first)
NORMal Normal byte order (MSB first)</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Format Data Command

Format Data Query</td><td>Sets or returns the data format setting. This is the format that is used for waveform data for all waveforms. The ZT5210 supports INT16 and REAL32 data formats only.

Recommendations:
Use INT16 for all waveforms. This is the fastest mode since the waveforms are stored as 16-bit integers. It also preserves the data resolution and accuracy.

Command Syntax
FORMat:DATA <format>,<bits>

Query Syntax
FORMat:DATA? → <format>,<bits>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th colspan="2">Range</th></tr>
<tr><td><format></td><td>Discrete</td><td>INTeger
REAL</td><td>Integers
Real numbers</td></tr>
<tr><td><bits></td><td>Discrete</td><td>16
32</td><td>16 bit
32 bit</td></tr>
</table>
</td></tr>
<tr><td>Format Precision Command

Format Precision Query</td><td>Sets or returns the precision format setting. This precision is used by the instrument for all floating point value returns. The precision is described using the format:

precision = mantissa^exponent

Command Syntax
FORMat:PRECision <mantissa>,<exponent>

Query Syntax
FORMat:PRECision? → <mantissa>,<exponent>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><mantissa></td><td>Integer</td><td>2-16</td></tr>
<tr><td><exponent></td><td>Integer</td><td>1-4</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Frequency Command

Frequency Query</td><td>
Sets or queries the selected channel's frequency. For amplitude and frequency modulated waveforms this is the carrier frequency. For multi-tone waveforms this parameter is query only and returns the total tone frequency. For serial data waveforms this parameter is the word frequency.

Command Syntax
[SOURce<n>:]FREQuency[:CW] <freq>

Query Syntax
[SOURce<n>:]FREQuency[:CW]? → <freq>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><freq></td><td>Float</td><td>Frequency in Hertz.
Range: See table below</td></tr>
</table>
Range
<table>
<tr><th>Shape</th><th>Minimum</th><th>Maximum</th></tr>
<tr><td>Sine
Haversine
Havercosine
Half Cycle Sine
Sinc Pulse
Lorentz Pulse</td><td>1 mHz</td><td>50 MHz</td></tr>
<tr><td>Square
Triangle
Ramp Up/Down
Pulse
Gaussian Pulse</td><td>1 mHz</td><td>20 MHz</td></tr>
<tr><td>Periodic Random Noise</td><td>1 mHz</td><td>1 MHz</td></tr>
<tr><td>AM
FM
Multi-Tone</td><td>100 Hz</td><td>50 MHz</td></tr>
<tr><td>Serial Data</td><td>16 µHz</td><td>5 MHz</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Function Shape Command

Function Shape Query</td><td>Sets or queries the selected channel’s standard function shape.

Command Syntax
[SOURce<n>:]FUNCtion[:SHAPe] <shape>

Query Syntax
[SOURce<n>:]FUNCtion[:SHAPe]? → <shape>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><shape></td><td>Discrete</td><td>SINE Sine
SQUare Square
TRIangle Triangle
RUP Ramp Up
RDOWn Ramp Down
DC DC
HSINe Haversine
HCOSine Havercosine
HCSine Half Cycle Sine
PULSe Pulse
SINC Sinc Pulse
GAUSsian Gaussian Pulse
LORentz Lorentz Pulse
PRNoise Periodic Random Noise
AM Amplitude Modulation
FM Frequency Modulation
MTONe Multi-Tone
SDATa Serial Data</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Gaussian Standard Deviation Command

Gaussian Standard Deviation Query</td><td>Sets or queries the standard deviation value used to generate Gaussian pulses on the selected channel. Gaussian pulses are defined as follows, where σ is the standard deviation:

$gauss(t) = e^{[-(t/\sigma)^2]}$

Command Syntax
[SOURce<n>:]GAUSsian:SDEViation <std_dev>

Query Syntax
[SOURce<n>:]GAUSsian:SDEViation? → <std_dev>

Parameters
<table><tr><th>Name</th><th>Type</th><th>Range</th></tr><tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr><tr><td><std_dev></td><td>Float</td><td>0 to ½ Period</td></tr></table>
Resolution
One DAC clock cycle</td></tr>
<tr><td>Initiate Continuous Command

Initiate Continuous Query</td><td>Sets or returns the instrument initiate continuous state. This is only used for Burst Mode.

Command Syntax
INITiate:CONTinuous <state>

Query Syntax
INITiate:CONTinuous? → <state>

Parameters
<table><tr><th>Name</th><th>Type</th><th>Range</th></tr><tr><td><state></td><td>Discrete</td><td>ON or 1 Initiate continuous ON
OFF or 0 Initiate continuous OFF</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Initiate Command

Initiate Query</td><td>Initiates the instrument. While initiated, the instrument is enabled to output waveforms. The query returns the instrument's initiation state.

Command Syntax
INITiate[:IMMediate]

Query Syntax
INITiate? → <state>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><state></td><td>Discrete</td><td>1 Instrument Initiated
0 Instrument Not Initiated</td></tr>
</table>
</td></tr>
<tr><td>Lorentz Half Width Command

Lorentz Half Width Query</td><td>Sets or queries the half width value used to generate Lorentz pulses on the selected channel. Lorentz pulses are defined as follows, where σ is the half width:

$Lorentz(t) = 1/[1+(t/\sigma)^2]$

Command Syntax
[SOURce<n>:]LORentz:HWIDth <width>

Query Syntax
[SOURce<n>:]LORentz:HWIDth? → <width>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><width></td><td>Float</td><td>0 to ½ Period</td></tr>
</table>
Resolution
One DAC clock cycle</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Multi-Tone Default Command</td><td>Resets the channel’s multi-tone settings to default.

Command Syntax
[SOURce<n>:]MTONe:DEFault

Query Syntax
None

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
</table></td></tr>
<tr><td>Multi-Tone Tone Frequency Command

Multi-Tone Tone Frequency Query</td><td>Sets or queries the frequency of a tone that is used for multi-tone on a channel. All active tones on a channel must be separated by an integer multiple of the minimum tone separation frequency.

Command Syntax
[SOURce<n>:]MTONe:TONE<t>:FREQuency <freq>

Query Syntax
[SOURce<n>:]MTONe:TONE<t>:FREQuency? → <freq>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><t></td><td>Integer</td><td>1 to 16</td></tr>
<tr><td><freq></td><td>Float</td><td>Frequency in Hertz: 100 Hz to 50 MHz</td></tr>
</table>
Resolution
100 Hz</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Multi-Tone Tone State Command

Multi-Tone Tone State Query</td><td>Sets or queries the state of a tone that is used for multi-tone on a channel. All active tones on a channel will be used to create a multi-tone waveform regardless of order.

Command Syntax
[SOURce<n>:]MTONe:TONE<t>[:STATe] <state>

Query Syntax
[SOURce<n>:]MTONe:TONE<t>[:STATe]? → <state>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><t></td><td>Integer</td><td>1 to 16</td></tr>
<tr><td><state></td><td>Discrete</td><td>ON or 1 Tone On
OFF or 0 Tone Off</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Operation Mode Command

Operation Mode Query</td><td>Sets or queries the channel’s operation mode.

• Two channels may not be set to Burst and Binary Modulation at the same time.
• All channels in Burst Mode use the same triggering information.
• All channels in Binary Modulation Mode use the same modulation source.
• All channels in Sweep Mode use the same Sweep Time.

Command Syntax
[SOURce<n>:]OPERation:MODE <mode>

Query Syntax
[SOURce<n>:]OPERation:MODE? → <mode>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><mode></td><td>Discrete</td><td>CONTinuous Continuous
BURSt Burst
SWEep Sweep
BMODulation Binary Modulation</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output ECL Trigger Polarity Command

Output ECL Trigger Polarity Query</td><td>
Sets or queries the unit VXIbus ECLTn output polarity. VXI only. The following considerations apply:

When positive output polarity is selected, the VXIbus output driver will force logic 1 onto the ECLT output when the signal source is active; i.e. for a POSitive polarity setting and an ARM source signal, the ECLT line will be logic 1 when the unit is armed.
Output polarity does not affect VXIbus ECLT line sensing used by other unit functions.
Each ECLTn output line polarity is selected individually
ECLT lines can be sourced and sensed simultaneously.

Command Syntax
OUTPut:ECLTrg<n>:POLarity <polarity>

Query Syntax
OUTPut:ECLTrg<n>:POLarity? →<polarity>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>0 ECLT0
1 ECLT1</td></tr>
<tr><td><polarity></td><td>Discrete</td><td>POSitive Positive polarity
NEGative Negative polarity</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output ECL Trigger Source Command

Output ECL Trigger Source Query</td><td>Sets or queries the unit VXIbus ECLTn output source. VXI only. The following considerations apply:

• The ECLTn output driver may be enabled or disabled and the output polarity selected; refer to the Output ECL Trigger Polarity Command.
• Each ECLTn output line source is selected individually.
• ECLT lines can be sourced and sensed simultaneously.

Command Syntax
OUTPut:ECLTrg<n>:SOURce <source>

Query Syntax
OUTPut:ECLTrg<n>:SOURce? →<source>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>0 ECLT0
1 ECLT1</td></tr>
<tr><td><source></td><td>Discrete</td><td>ARM Arm Event
TRIGger Trigger Event
GCOMplete Generation Complete Event
CONStant Constant
OPC Operation complete Event
MSS Master Status Summ. Event
SYNC<n> Sync Pulse, where <n> may be 1, 2, 3 or 4.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output ECL Trigger State Command

Output ECL Trigger State Query</td><td>Sets or queries the unit VXIbus ECLTn line state. VXI only. The following considerations apply:

• The ECLTn output source and polarity are selectable.
• Each ECLTn line state is selected individually.
• ECLT lines can be sourced and sensed simultaneously.

Command Syntax
OUTPut:ECLTrg<n>[:STATe] <state>

Query Syntax
OUTPut:ECLTrg<n>[:STATe]? →<state>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>0 ECLT0
1 ECLT1</td></tr>
<tr><td><state></td><td>Discrete</td><td>ON or 1 Enable
OFF or 0 Disable</td></tr>
</table>
</td></tr>
<tr><td>Output Event Time Command

Output Event Time Query</td><td>Sets or queries the output event time. This is the length of time that event driven output pulses will stay high. The time is given in seconds.

Command Syntax
OUTPut:EVENt[:TIME] <seconds>

Query Syntax
OUTPut:EVENt[:TIME]? → <seconds>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><seconds></td><td>Float</td><td>Time in seconds.
MINimum 50 ns
MAXimum 0.163 s</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output External Polarity Command

Output External Polarity Query</td><td>Sets or queries the polarity of the external output.
When the external output source is CONStant, positive polarity outputs a constant high, negative polarity outputs a constant low.
When the external output source is a clock or pulse, positive polarity outputs a low signal with a high pulse; negative polarity outputs a high signal with a low pulse.
Command Syntax
OUTPut:EXTernal:POLarity <polarity>

Query Syntax
OUTPut:EXTernal:POLarity?→<polarity>

Parameters<table><tr><td>Name</td><td>Type</td><td colspan="2">Range</td></tr><tr><td><polarity></td><td>Discrete</td><td>POSitive
NEGative</td><td>Positive polarity
Negative polarity</td></tr></table></td></tr>
<tr><td>Output External Pulse Mode Command

Output External Pulse Mode Query</td><td>Sets or queries the external output pulse mode. Pulse mode outputs a 16.667 ns wide pulse. Clock mode outputs a 50% duty cycle clock.

Command Syntax
OUTPut:EXTernal:PULSe:MODE <mode>

Query Syntax
OUTPut:EXTernal:PULSe:MODE?→<mode>

Parameters<table><tr><td>Name</td><td>Type</td><td colspan="2">Range</td></tr><tr><td><mode></td><td>Discrete</td><td>PULSe
CLOCk</td><td>Pulse Mode
Clock Mode</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output External Pulse Period Command

Output External Pulse Period Query</td><td>Sets or queries the external output pulse period. The instrument supports a programmable interval from 26.667 ns to 100 seconds.

Command Syntax
OUTPut:EXTernal:PULSe:PERiod <period>

Query Syntax
OUTPut:EXTernal:PULSe:PERiod?→<period>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><period></td><td>Float</td><td>Time in seconds
MINimum 26.667 ns
MAXimum 100 s</td></tr>
</table>
</td></tr>
<tr><td>Output External Source Command

Output External Source Query</td><td>Sets or queries the external output source

Command Syntax
OUTPut:EXTernal:SOURce <source>

Query Syntax
OUTPut:EXTernal:SOURce?→<source>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td colspan="2">Range</td></tr>
<tr><td><source></td><td>Discrete</td><td>ARM</td><td>Arm Event</td></tr>
<tr><td></td><td></td><td>TRIGger</td><td>Trigger Complete Event</td></tr>
<tr><td></td><td></td><td>GCOMplete</td><td>Generation Complete Event</td></tr>
<tr><td></td><td></td><td>OPC</td><td>Operation Complete Event</td></tr>
<tr><td></td><td></td><td>MSS</td><td>Master Status Summ. Event</td></tr>
<tr><td></td><td></td><td>CONStant</td><td>Constant State</td></tr>
<tr><td></td><td></td><td>REFerence</td><td>Reference Clock</td></tr>
<tr><td></td><td></td><td>PULSe</td><td>Pulse</td></tr>
<tr><td></td><td></td><td>CCLock</td><td>Common Clock / 2</td></tr>
<tr><td></td><td></td><td>SYNC<n></td><td>Sync Pulse, where <n> may be 1, 2, 3 or 4.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output External State Command

Output External State Query</td><td>Sets or queries the external output state

Command Syntax
OUTPut:EXTernal[:STATe] <state>

Query Syntax
OUTPut:EXTernal[:STATe]?→<state>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><state></td><td>Discrete</td><td>ON or 1 Active state
OFF or 0 Inactive state</td></tr>
</table>
</td></tr>
<tr><td>Output LXI Mode Command

Output LXI Mode Query</td><td>Sets or queries the unit LXIn output mode. LXI only.

Command Syntax
OUTPut:LXI<n>:MODE <mode>

Query Syntax
OUTPut:LXI<n>:MODE?→<mode>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>LXI output line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7</td></tr>
<tr><td><mode></td><td>Discrete</td><td>WIRedor Wired Or
DRIVen Driven
BIAS Bias</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output LXI Polarity Command

Output LXI Polarity Query</td><td>Sets or queries the unit LXIn output polarity. LXI only. This command and query have identical functionality to Output TTL Trigger Polarity command and query. The following considerations apply:

• When positive output polarity is selected, output driver will force logic 1 onto the output when the signal source is active; i.e. for a POSitive polarity setting and an ARM source signal, the line will be logic 1 when the unit is armed.
• Output polarity does not affect line sensing used by other unit functions.
• Each LXIn output line polarity is selected individually.
• Output lines can be sourced and sensed simultaneously.

Command Syntax
OUTPut:LXI<n>:POLarity <polarity>

Query Syntax
OUTPut:LXI<n>:POLarity? →<polarity>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>LXI output line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7</td></tr>
<tr><td><polarity></td><td>Discrete</td><td>POSitive Positive polarity
NEGative Negative polarity</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output LXI Source Command

Output LXI Source Query</td><td>Sets or queries the unit LXIn output source. This command and query have identical functionality to Output TTL Trigger Source command and query. The following considerations apply:

• The LXIn line may be enabled or disabled and the output polarity selected.
• Each LXIn output line source is selected individually.
• LXI lines can be sourced and sensed simultaneously.

Command Syntax
OUTPut:LXI<n>:SOURce <source>

Query Syntax
OUTPut:LXI<n>:SOURce? → <source>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>LXI output line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7</td></tr>
<tr><td><source></td><td>Discrete</td><td>ARM Arm Event
TRIGger Trigger Complete Event
GCOMplete Generation Complete Event
OPC Operation Complete Event
MSS Master Status Summ. Event
CONStant Constant State
SYNC<n> Sync Pulse, where <n> may be 1, 2, 3 or 4.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output LXI State Command

Output LXI State Query</td><td>Sets or queries the unit LXIn line state. This command and query have identical functionality to Output TTL Trigger State command and query. The following considerations apply:

• The LXIn output source and polarity are selectable.
• Each LXIn line state is selected individually.
• LXI lines can be sourced and sensed simultaneously.

Command Syntax
OUTPut:LXI<n>[:STATe] <state>

Query Syntax
OUTPut:LXI<n>[:STATe]? → <state>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><n></td><td>Discrete</td><td>LXI output line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7</td></tr><tr><td><state></td><td>Discrete</td><td>ON or 1 Active state
OFF or 0 Inactive state</td></tr></table></td></tr>
<tr><td>Output Mode Command

Output Mode Query</td><td>Sets or queries the channel's output mode. In Standard Mode the instrument will recalculate a new waveform with automatically coerced parameters based on the selected channel shape and last changed parameter. In Arbitrary Mode and Sequence mode, the instrument will apply the requested settings to the channel without modifying the waveform codes.

Command Syntax
[SOURce<n>:]OUTPut:MODE <mode>

Query Syntax
[SOURce<n>:]OUTPut:MODE? → <mode>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr><tr><td><mode></td><td>Discrete</td><td>FUNCtion Standard Function
ARBitrary Arbitrary Waveform
SEQuence Arbitrary Sequence</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output Source Query</td><td>Queries the Output Channel Source. The source is set whenever a waveform is copied to the channel or a sequence is generated to the channel. If the data codes in the channel have been generated using a standard function or uploaded, the channel will return itself as the source.

Command Syntax
None

Query Syntax
OUTP<n>:SOURce? → <source>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><source></td><td>Discrete</td><td>OUTPut<n> Output Channel, where <n> may be 1, 2, 3 or 4
REFerence<n> Reference Channel, where <n> may 1, 2, 3 or 4.
WAVeform<n> Waveform Library location where <n> is the handle.
SEQuence<n> Sequence where <n> may be 1, 2, 3, 4, 5, 6, 7 or 8.</td></tr>
</table></td></tr>
<tr><td>Output State Command

Output State Query</td><td>Enables or disables an output channel generation. To generate waveforms the channel must be enabled and the instrument must also be initiated.

Command Syntax
OUTPut<n>[:STATe] <state>

Query Syntax
OUTPut<n>[:STATe]? → <state>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><state></td><td>Discrete</td><td>ON or 1 Input channel generation ON
OFF or 0 Input channel generation OFF</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output Sync Off Position Command

Output Sync Off Position Query</td><td>Sets or queries the unit SYNCn Off Position. Output channels that have SYNCn as a source will be active during the time the instrument is generating OUTPn between the On and Off Positions.

Command Syntax
OUTPut:SYNC<n>:OFF:POSition <position>

Query Syntax
OUTPut:SYNC<n>:OFF:POSition? → <position>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>SYNC bit channel, where <n> may be 1, 2, 3 or 4</td></tr>
<tr><td><position></td><td>Float</td><td>Fraction of total waveform length
MINimum 0.0
MAXimum 1.00
May also be entered as a percentage: 0PCT to 100PCT</td></tr>
</table>
</td></tr>
<tr><td>Output Sync On Position Command

Output Sync On Position Query</td><td>Sets or queries the unit SYNCn On Position. Output channels that have SYNCn as a source will be active during the time the instrument is generating OUTPn between the On and Off Positions.

Command Syntax
OUTPut:SYNC<n>:ON:POSition <position>

Query Syntax
OUTPut:SYNC<n>:ON:POSition? → <position>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>SYNC bit channel, where <n> may be 1, 2, 3 or 4</td></tr>
<tr><td><position></td><td>Float</td><td>Fraction of total waveform length
MINimum 0.0
MAXimum 1.00
May also be entered as a percentage: 0PCT to 100PCT</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output TTL Trigger Polarity Command

Output TTL Trigger Polarity Query</td><td>Sets or queries the unit bus TTLTn line output polarity. The following considerations apply:

• When positive output polarity is selected, the output driver will force logic 1 onto the TTLT output when the signal source is active, i.e. for a positive polarity setting and an Arm source signal, the TTLT line will be logic 1 when the unit is armed.
• Output polarity does not affect the bus TTLT line sensing used by other unit functions.
• Each output line polarity is selected individually.
• TTLT lines can be sourced and sensed simultaneously.

Command Syntax
OUTPut:TTLTrg<n>:POLarity <polarity>

Query Syntax
OUTPut:TTLTrg<n>:POLarity? → <polarity>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><n></td><td>Discrete</td><td>Bus TTLT line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7</td></tr><tr><td><polarity></td><td>Discrete</td><td>NEGative Negative polarity
POSitive Positive polarity</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output TTL Trigger Source Command

Output TTL Trigger Source Query</td><td>Sets or queries the unit TTLTn output driver source. The following considerations apply:

• The TTLT line may be enabled or disabled and the polarity selected.
• Each line output source is selected individually.
• TTLT lines can be sourced and sensed simultaneously.

Command Syntax
OUTPut:TTLTrg<n>:SOURce <source>

Query Syntax
OUTPut:TTLTrg<n>:SOURce? → <source>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>Bus TTLT line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7</td></tr>
<tr><td><source></td><td>Discrete</td><td>ARM Arm Event
TRIGger Trigger Complete Event
GCOMplete Generation Complete Event
OPC Operation Complete Event
MSS Master Status Summ. Event
CONStant Constant State
SYNC<n> Sync Pulse, where <n> may be 1, 2, 3 or 4.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Output TTL Trigger State Command

Output TTL Trigger State Query</td><td>Sets or queries the unit TTLTn state. The following considerations apply:

• The TTLT output source and polarity are selectable.
• Each TTLT line state is selected individually.
• TTLT lines can be sourced and sensed simultaneously.

Command Syntax
OUTPut:TTLTrg<n>[:STATe] <state>

Query Syntax
OUTPut:TTLTrg<n>[:STATe]? → <state>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>Bus TTLT line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7</td></tr>
<tr><td><state></td><td>Discrete</td><td>ON or 1 Active state
OFF or 0 Inactive state</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Period Command

Period Query</td><td>
Sets or queries the channel's period. For amplitude and frequency modulated waveforms this is the carrier period. For multi-tone waveforms this parameter is query only and returns the total tone period. For serial data waveforms this parameter is the word period.

Command Syntax
[SOURce<n>:]PERiod <time>

Query Syntax
[SOURce<n>:]PERiod? → <time>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><time></td><td>Float</td><td>Period in seconds
Range: See table below</td></tr>
</table>
Range
<table>
<tr><th>Shape</th><th>Minimum</th><th>Maximum</th></tr>
<tr><td>Sine
Haversine
Havercosine
Half Cycle Sine
Sinc Pulse
Lorentz Pulse
Multi-Tone</td><td>20 ns</td><td>1000 s</td></tr>
<tr><td>Square
Triangle
Ramp Up/Down
Pulse
Gaussian Pulse</td><td>50 ns</td><td>1000 s</td></tr>
<tr><td>Periodic Random Noise</td><td>1 µs</td><td>1000 s</td></tr>
<tr><td>AM
FM
Multi-Tone</td><td>20 ns</td><td>10 ms</td></tr>
<tr><td>Serial Data</td><td>200 ns</td><td>64 ks</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Phase Command

Phase Query</td><td>Sets or queries the channel’s output phase adjustment.

Command Syntax
[SOURce<n>:]PHASe[:ADJust] <phase>

Query Syntax
[SOURce<n>:]PHASe[:ADJust]? → <phase>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><phase></td><td>Float</td><td>Phase in Radians
MINimum 0 rad
MAXimum 2π rad
May also be entered in degrees:
0DEG to 360DEG</td></tr>
</table>
</td></tr>
<tr><td>Pulse Transition Leading Command

Pulse Transition Leading Query</td><td>Sets or queries the channel’s pulse leading transition time. This value is only used when the channel’s Function Shape is set to Pulse.

Command Syntax
[SOURce<n>:]PULSe:TRANsition[:LEADing] <time>

Query Syntax
[SOURce<n>:]PULSe:TRANsition[:LEADing]? → <time>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><time></td><td>Float</td><td>Transition time in seconds:
0 to period</td></tr>
</table>
Resolution
One DAC clock cycle</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Pulse Transition Trailing Command

Pulse Transition Trailing Query</td><td>Sets or queries the channel's pulse trailing transition time. This value is only used when the channel's Function Shape is set to Pulse.

Command Syntax
[SOURce<n>:]PULSe:TRANsition:TRAiling <time>

Query Syntax
[SOURce<n>:]PULSe:TRANsition:TRAiling? → <time>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><time></td><td>Float</td><td>Transition time in seconds:
0 to period</td></tr>
</table>
Resolution
One DAC clock cycle</td></tr>
<tr><td>Reference Oscillator Frequency Query</td><td>Queries the frequency of the reference clock that provides the instrument time base.

Command Syntax
None

Query Syntax
[SOURce:]ROSCillator:FREQuency? → <freq>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><freq></td><td>Float</td><td>Frequency in Hertz</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Reference Oscillator Output State Command

Reference Oscillator Output State Query</td><td>Sets or queries the state of the reference clock output. PCI only.

Command Syntax
[SOURce:]ROSCillator:OUTPut[:STATe] <state>

Query Syntax
[SOURce:]ROSCillator:OUTPut[:STATe]? → <state>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><state></td><td>Discrete</td><td>ON or 1 Active state
OFF or 0 Inactive state</td></tr>
</table>
</td></tr>
<tr><td>Reference Oscillator Source Command

Reference Oscillator Source Query</td><td>Sets or queries the source for the reference clock that provides the instrument time base.

Command Syntax
[SOURce:]ROSCillator:SOURce <source>

Query Syntax
[SOURce:]ROSCillator:SOURce? → <source>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><source></td><td>Discrete</td><td>INTernal Local reference
EXTernal External input
CLK10 Clock 10 / Backplane</td></tr>
</table>
</td></tr>
<tr><td>Sense ECL Trigger State Query</td><td>Queries the unit’s VXIbus ECLTn output driven state. VXI only.

Command Syntax
None

Query Syntax
SENSe:ECLTrg<n>[:STATe]? → <state>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>0 ECLT0
1 ECLT1</td></tr>
<tr><td><state></td><td>Discrete</td><td>ON or 1 Active state
OFF or 0 Inactive state</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Sense LXI State Query</td><td>Queries the unit LXIn output driven state. This query has identical functionality to Sense TTL Trigger State Query. LXI only.

Command Syntax
None

Query Syntax
SENSe:LXI<n>[:STATe]? → <state>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><n></td><td>Discrete</td><td>LXI output line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7</td></tr><tr><td><state></td><td>Discrete</td><td>ON or 1 Active state
OFF or 0 Inactive state</td></tr></table></td></tr>
<tr><td>Sense TTL Trigger State Query</td><td>Queries the unit’s TTLTn output driven state.

Command Syntax
None

Query Syntax
SENSe:TTLTrg<n>[:STATe]? → <state>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><n></td><td>Discrete</td><td>TTLT line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7</td></tr><tr><td><state></td><td>Discrete</td><td>ON or 1 Active state
OFF or 0 Inactive state</td></tr></table></td></tr>
<tr><td>Sequence Clear Command</td><td>Resets the specified sequence’s information to the default state.

Command Syntax
[SOURce:]SEQuence:CLEar <seq_handle>

Query Syntax
None

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><seq_handle></td><td>Integer</td><td>1-8</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Sequence Clear All Command</td><td>Resets all sequence information to the default state.

Command Syntax
[SOURce:]SEQuence:CLEar:ALL

Query Syntax
None

Parameters
None</td></tr>
<tr><td>Sequence Data Loop Count Command

Sequence Data Loop Count Query</td><td>Sets or queries the loop count for a sequence stage.

Command Syntax
[SOURce:]SEQuence:DATA:LCOunt <seq_handle>,<stage>,<count>

Query Syntax
[SOURce:]SEQuence:DATA:LCOunt? <seq_handle>,<stage> → <count>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><seq_handle></td><td>Integer</td><td>1-8</td></tr>
<tr><td><stage></td><td>Integer</td><td>1-4096</td></tr>
<tr><td><count></td><td>Integer</td><td>1-65535</td></tr>
</table></td></tr>
<tr><td>Sequence Data Waveform Command

Sequence Data Waveform Query</td><td>Sets or queries the source waveform location for a sequence stage.

Command Syntax
[SOURce:]SEQuence:DATA:WAVeform <seq_handle>,<stage>,<wave_source>

Query Syntax
[SOURce:]SEQuence:DATA:WAVeform? <seq_handle>,<stage> → <wave_source>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><seq_handle></td><td>Integer</td><td>1-8</td></tr>
<tr><td><stage></td><td>Integer</td><td>1-4096</td></tr>
<tr><td><wave_source></td><td>Discrete</td><td>REFerence<n> Reference Channel, where <n> may 1, 2, 3 or 4.
WAVeform<n> Waveform Library location where <n> is the handle.</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Sequence Generate Query</td><td>Generates a sequence to the specified output channel.

Command Syntax
None

Query Syntax
[SOURce<n>:]SEQuence:GENerate? <seq_handle> → <status>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><seq_handle></td><td>Integer</td><td>1-8</td></tr>
<tr><td><status></td><td>Discrete</td><td>1 Generation successful
0 Generation failed</td></tr>
</table></td></tr>
<tr><td>Sequence Loop Maximum Query</td><td>Queries the maximum loop count that may be set for a sequence stage.

Command Syntax
None

Query Syntax
[SOURce:]SEQuence:LOOP:MAXimum? → <count>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><count></td><td>Integer</td><td>65535</td></tr>
</table></td></tr>
<tr><td>Sequence Maximum Query</td><td>Queries the maximum number of sequences that can be stored in instrument memory.

Command Syntax
None

Query Syntax
[SOURce:]SEQuence:MAXimum? → <max>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><max></td><td>Integer</td><td>8</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Sequence Size Command

Sequence Size Query</td><td>Sets or queries the number of stages to use for the specified sequence. Sequences are generated from consecutive stages from 1 to Sequence Size.

Command Syntax
[SOURce:]SEQuence:SIZE <seq_handle>,<size>

Query Syntax
[SOURce:]SEQuence:SIZE? <seq_handle> → <size>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><seq_handle></td><td>Integer</td><td>1-8</td></tr><tr><td><size></td><td>Integer</td><td>2-4096</td></tr></table></td></tr>
<tr><td>Sequence Size Maximum Query</td><td>Queries the maximum number of stages that sequences can have.

Command Syntax
None

Query Syntax
[SOURce:]SEQuence:SIZE:MAXimum? → <max>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><max></td><td>Integer</td><td>4096</td></tr></table></td></tr>
<tr><td>Sequence Size Minimum Query</td><td>Queries the minimum number of stages that sequences can have.

Command Syntax
None

Query Syntax
[SOURce:]SEQuence:SIZE:MINimum? → <min>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><min></td><td>Integer</td><td>2</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Serial Data Bit Period Command

Serial Data Bit Period Query</td><td>Sets or queries the channel's serial data bit period. This period is the amount of time used to play each bit of a serial data word. The total word play time can be set or queried using Period Command/Query.

Command Syntax
[SOURce<n>:]SDATa:BPERiod <time>

Query Syntax
[SOURce<n>:]SDATa:BPERiod? → <time>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><time></td><td>Float</td><td>Bit period in seconds.
MINimum 50 ns
MAXimum 1000 s</td></tr>
</table>
</td></tr>
<tr><td>Serial Data Word Length Command

Serial Data Word Length Query</td><td>Sets or queries the channel's serial data word length. The word length is the number of bits that make up a serial data word. The output word may be truncated based on the word length.

Command Syntax
[SOURce<n>:]SDATa:WLENgth <bits>

Query Syntax
[SOURce<n>:]SDATa:WLENgth? → <bits>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><bits></td><td>Integer</td><td>4 to 64</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Serial Data Word Command

Serial Data Word Query</td><td>Sets or queries the channel's serial data word. The output word may be truncated based on the word length.

Command Syntax
[SOURce<n>:]SDATa:WORD <word>

Query Syntax
[SOURce<n>:]SDATa:WORD? → <word>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><word></td><td>64-bit Unsigned Integer</td><td>0 to 2^{64}-1
(0x0000000000000000 to
0xFFFFFFFFFFFFFFFF)</td></tr>
</table>
</td></tr>
<tr><td>Sinc Frequency Command

Sinc Frequency Query</td><td>Sets or queries the channel's sinc frequency. The ratio of the sinc frequency to the function frequency must be less than or equal to 100,000:1. The sinc frequency, f, is used to create Sinc Pulse outputs, where sinc is defined as:

$$\text{sinc}(t) = \sin(2\pi f*t)/(2\pi f*t)$$

Command Syntax
[SOURce<n>:]SINC:FREQuency <freq>

Query Syntax
[SOURce<n>:]SINC:FREQuency? → <freq>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><freq></td><td>Float</td><td>Function frequency to 50 MHz</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Interrupt Request State Command

Status Interrupt Request State Query</td><td>Sets or queries the interrupt request state.

Command Syntax
STATus:IRQ[:STATe] <state>

Query Syntax
STATus:IRQ[:STATe]? → <freq>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th colspan="2">Range</th></tr>
<tr><td><state></td><td>Discrete</td><td>ON or 1
OFF or 0</td><td>Active state
Inactive state</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Operation Condition Query</td><td>
Queries the contents of the Operation Status Condition Register. The Operation Status Condition Register identifies currently running processes, such as waveform generation. The following considerations apply when using the Status Operation Condition Query:

The Operation Status Condition Register identifies current running processes. Use the Status Operation Event Query to identify a history of which processes have run since the last operation event status check.
The Status Operation Condition Query does not clear the Operation Status Condition Register.

Command Syntax
None

Query Syntax
STATus:OPERation:CONDition? → <condition>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><condition></td><td>16-bit mask</td><td>0 to 65535
Bit 0 Unused
Bit 1 Settling
Bit 2 Ranging
Bits 3-4 Unused
Bit 5 Waiting for trigger
Bit 6 Waiting for arm
Bit 7 Unused
Bit 8 Output 1 ON
Bit 9 Output 2 ON
Bit 10 Output 3 ON
Bit 11 Output 4 ON
Bit 12 Trigger Event
Bits 13–15 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Operation Enable Command

Status Operation Enable Query</td><td>Sets or queries the contents of the Operation Status Enable Register. The Operation Status Enable Register enables the reporting of operation events to the Status Byte. The following considerations apply when using the Status Operation Enable Command/Query:

• The Operation Status Enable Register is a bit mask that allows selected operation status events to be reported to the Status Byte.
• Only low to high (inactive to active) Operation Status Event Register bit transitions are reported.
• Operation status events report in bit 7 of the Status Byte.
• The Status Preset Command sets all register bits to zero (0), which disables all operation event reporting.

Command Syntax
STATus:OPERation:ENABle <enable>

Query Syntax
STATus:OPERation:ENABle? → <enable>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><enable></td><td>16-bit mask</td><td>0 to 65535
Bit 0 Unused
Bit 1 Settling
Bit 2 Ranging
Bits 3-4 Unused
Bit 5 Waiting for trigger
Bit 6 Waiting for arm
Bit 7 Unused
Bit 8 Output 1 ON
Bit 9 Output 2 ON
Bit 10 Output 3 ON
Bit 11 Output 4 ON
Bit 12 Trigger Event
Bits 13–15 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Operation Event Query</td><td>
Queries the contents of the Operation Status Event Register. The Operation Status Event Register identifies unit processes that have been run, such as waveform generation. The following considerations apply when using the Status Operation Event Query:

The Operation Status Event Register records the history of the processes that have been run since the previous operation event status query. Use the Status Operation Condition Query to identify currently running processes.
The Status Operation Event Query clears the Operation Status Event Register after returning the current register contents.
Operation Status Event Register bits may be summarized in the Status Byte.
In order to identify which processes have run between two times, ex. acquisition start and later status check, the Operation Event Status Register must be cleared by reading it before the processes are run.

Command Syntax
None

Query Syntax
STATus:OPERation[:EVENt]? → <event>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><event></td><td>16-bit mask</td><td>0 to 65535
Bit 0 Unused
Bit 1 Settling
Bit 2 Ranging
Bits 3-4 Unused
Bit 5 Waiting for trigger
Bit 6 Waiting for arm
Bit 7 Unused
Bit 8 Output 1 ON
Bit 9 Output 2 ON
Bit 10 Output 3 ON
Bit 11 Output 4 ON
Bit 12 Trigger Event
Bits 13–15 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Preset Command</td><td>Sets the status reporting event enable data structures to a known state. The condition and event register contents are not affected. All device-dependent status registers which cascade events into the Questionable Status and the Operation Status Registers are enabled by setting those device-dependent event enable registers to $7FFF_{16}$ (the 15 LSBs set). The IEEE-488.2 mandatory status data structures are disabled by setting the Questionable Status and Operation Status event enable registers to 0000_{16}. The Status Byte and Standard Event Status Registers as defined by IEEE 488.2 are not affected.

Command Syntax
STATus:PRESet

Query Syntax
None

Parameters
None</td></tr>
<tr><td>Status Questionable Calibration Condition Query</td><td>Queries the contents of the Questionable Calibration Status Condition Register. The Questionable Calibration Status Condition Register identifies current questionable results from running calibration processes. The following considerations apply when using the Status Questionable Calibration Condition Query:

• The Questionable Calibration Status Condition Register identifies current questionable results from running processes. Use the Status Questionable Calibration Event Query to identify which questionable results generated since the last questionable event status check.
• The Status Questionable Calibration Condition Query does <u>not</u> clear the Questionable Calibration Status Condition Register.

Command Syntax
None

Query Syntax
STATus:QUEStionable:CALibration:CONDition? → <condition>

Parameters
<table><tr><th>Name</th><th>Type</th><th>Range</th></tr><tr><td><condition></td><td>8-bit mask</td><td>0 to 255
Bit 0 Calibration Storage
Bits 1–7 Unused</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Calibration Enable Command

Status Questionable Calibration Enable Query</td><td>Sets or queries the contents of the Questionable Calibration Status Enable Register. The Questionable Calibration Status Enable Register enables the reporting of questionable events to the Status Byte. The following considerations apply when using the Status Questionable Calibration Enable Command/Query:

• The Questionable Calibration Status Enable Register is a bit mask that allows selected questionable status events to be reported to the Questionable Summary Register.
• Only low to high (inactive to active) Questionable Calibration Status Event Register bit transitions are reported.
• Questionable calibration status events are reported in bit 8 of the Questionable Summary Register.
• Status Preset Command sets all register bits to zero (0) which disables all questionable event reporting.

Command Syntax
STATus:QUEStionable:CALibration:ENABle <enable>

Query Syntax
STATus:QUEStionable:CALibration:ENABle? → <enable>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><enable></td><td>8-bit mask</td><td>0 to 255
Bit 0 Calibration Storage
Bits 1–7 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Calibration Event Query</td><td>Queries the Questionable Calibration Status Event Register. The Questionable Calibration Frequency Status Event Register identifies calibration processes that have completed with questionable results. The following considerations apply when using the Status Calibration Event Query:

• The Questionable Calibration Status Event Register records the history of the questionable calibration process results generated since the previous Status Questionable Calibration Event Query.
• The Status Questionable Calibration Event Query clears the Questionable Calibration Event Register after returning the current register contents.
• Questionable frequency events are reported in bit 8 of the Questionable Summary Register.
• In order to identify questionable results from a particular process, the Questionable Calibration Status Event Register must be cleared by reading it before the processes are run.

Command Syntax
None

Query Syntax
STATus:QUEStionable:CALibration[:EVENt]? → <event>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><event></td><td>8-bit mask</td><td>0 to 255
Bit 0 Calibration Storage
Bits 1–7 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Condition Query</td><td>Queries the contents of the Questionable Status Condition Register. The Questionable Status Condition Register identifies current questionable results from running processes, such as self-test. The following considerations apply when using the Status Questionable Condition Query:

• The Questionable Status Condition Register identifies current questionable results from running processes. Use the Status Questionable Event Query to identify which questionable results generated since the last questionable event status check.
• The Status Questionable Condition Query does <u>not</u> clear the Questionable Status Condition Register.

Command Syntax
None

Query Syntax
STATus:QUEStionable:CONDition? → <condition>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><condition></td><td>16-bit mask</td><td>0 to 65535
Bit 0 Voltage
Bits 1–3 Unused
Bit 4 Temperature
Bit 5 Frequency
Bits 6–8 Unused
Bit 9 Test
Bits 10–15 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Enable Command

Status Questionable Enable Query</td><td>Sets or queries the contents of the Questionable Status Enable Register. The Questionable Status Enable Register enables the reporting of questionable events to the Status Byte. The following considerations apply when using the Status Questionable Enable Command/Query:

• The Questionable Status Enable Register is a bit mask that allows selected questionable status events to be reported to the Status Byte.
• Only low to high (inactive to active) Questionable Status Event Register bit transitions are reported.
• Questionable status events are reported in bit 3 of the Status Byte. Refer to the Status Byte query.
• Status Preset Command sets all register bits to zero (0) which disables all questionable event reporting.

Command Syntax
STATus:QUEStionable:ENABle <enable>

Query Syntax
STATus:QUEStionable:ENABle? → <enable>

Parameters
<table><tr><th>Name</th><th>Type</th><th>Range</th></tr><tr><td><enable></td><td>16-bit mask</td><td>0 to 65535
Bit 0 Voltage
Bits 1–3 Unused
Bit 4 Temperature
Bit 5 Frequency
Bits 6–8 Unused
Bit 9 Test
Bits 10–15 Unused</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Event Query</td><td>
Queries the Questionable Status Event Register. The Questionable Status Event Register identifies unit processes that have completed with questionable results, such as self test errors. The following considerations apply when using the Status Questionable Event Query:

The Questionable Status Event Register records the history of the questionable process results generated since the previous questionable event status query.
The Status Questionable Event Query clears the Questionable Status Event Register after returning the current register contents.
Questionable Status Event Register bits may be summarized in the Status Byte.
In order to identify questionable results from a particular process, the Questionable Status Event Register must be cleared by reading it before the processes are run.

Command Syntax
None

Query Syntax
STATus:QUEStionable[:EVENt]? → <event>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><event></td><td>16-bit mask</td><td>0 to 65535
Bit 0 Voltage bit
Bits 1–3 Unused
Bit 4 Temperature
Bit 5 Frequency bit
Bits 6–8 Unused
Bit 9 Test bit
Bits 10–15 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Frequency Condition Query</td><td>
Queries the contents of the Questionable Frequency Status Condition Register. The Questionable Frequency Status Condition Register identifies current questionable results from all internally-generated clock frequency conditions. The following considerations apply when using the Status Frequency Condition Query:

The Questionable Frequency Status Condition Register identifies current conditions. Use the Status Questionable Frequency Event Query to identify a history of which frequency conditions have failed since the last event status check.
The Status Questionable Frequency Condition Query does <u>not</u> clear the Questionable Frequency Status Condition Register.

Command Syntax
None

Query Syntax
STATus:QUEStionable:FREQuency:CONDition? → <condition>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><condition></td><td>16-bit mask</td><td>0 to 65535
Bit 0 PLL 1-2 Unlocked
Bit 1 PLL 3-4 Unlocked
Bit 2 Memory 1 Clock Unlocked
Bit 3 Memory 2 Clock Unlocked
Bit 4 Memory 3 Clock Unlocked
Bit 5 Memory 4 Clock Unlocked
Bits 6–14 Unused
Bit 15 Baseboard Clock Unlocked</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Frequency Enable Command

Status Questionable Frequency Enable Query</td><td>
Sets or queries the contents of the Questionable Frequency Status Enable Register. The Questionable Frequency Status Enable Register enables the reporting of questionable frequency events to the Questionable Summary Register. The following considerations apply when using the Status Frequency Enable Command/Query:

The Questionable Frequency Status Enable Register is a bit mask that allows selected questionable frequency events to be reported to the Questionable Summary Register.
Only low to high (inactive to active) Questionable Frequency Status Enable Register bit transitions are reported.
Questionable frequency events are reported in bit 5 of the Questionable Summary Register.
The Status Preset Command sets all 15 LSBs to one (1), which enables all event reporting.

Command Syntax
STATus:QUEStionable:FREQuency:ENABle <enable>

Query Syntax
STATus:QUEStionable:FREQuency:ENABle? → <enable>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><enable></td><td>16-bit mask</td><td>0 to 65535
Bit 0 PLL 1-2 Unlocked
Bit 1 PLL 3-4 Unlocked
Bit 2 Memory 1 Clock Unlocked
Bit 3 Memory 2 Clock Unlocked
Bit 4 Memory 3 Clock Unlocked
Bit 5 Memory 4 Clock Unlocked
Bits 6–14 Unused
Bit 15 Baseboard Clock Unlocked</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Frequency Event Query</td><td>
Queries the Questionable Frequency Status Event Register. The Questionable Frequency Status Event Register identifies frequency processes that have completed with questionable results. The following considerations apply when using the Status Frequency Event Query:

The Questionable Frequency Status Event Register records the history of the questionable frequency process results generated since the previous Status Questionable Frequency Event Query.
The Status Questionable Frequency Event Query clears the Questionable Frequency Event Register after returning the current register contents.
Questionable frequency events are reported in bit 5 of the Questionable Summary Register.
In order to identify questionable results from a particular process, the Questionable Frequency Status Event Register must be cleared by reading it before the processes are run.

Command Syntax
None

Query Syntax
STATus:QUEStionable:FREQuency[:EVENt]? → <event>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><event></td><td>16-bit mask</td><td>0 to 65535
Bit 0 PLL 1-2 Unlocked
Bit 1 PLL 3-4 Unlocked
Bit 2 Memory 1 Clock Unlocked
Bit 3 Memory 2 Clock Unlocked
Bit 4 Memory 3 Clock Unlocked
Bit 5 Memory 4 Clock Unlocked
Bits 6–14 Unused
Bit 15 Baseboard Clock Unlocked</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Test Condition Query</td><td>
Queries the contents of the Questionable Test Status Condition Register. The Questionable Test Status Condition Register identifies the test results of memory (RAM, DRAM, and Flash) tests, along with register and PLL tests. The following considerations apply when using the Status Questionable Test Condition Query:

The Questionable Test Status Condition Register identifies current tests. Use the Status Questionable Test Condition Query to identify a history of which tests have failed since the last test status check.
Questionable test events report in bit 9 of the Questionable Summary Register.
The Status Questionable Test Condition Query does not clear the Questionable Test Status Condition Register.

Command Syntax
None

Query Syntax
STATus:QUEStionable:TEST:CONDition? → <condition>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th><th></th></tr>
<tr><td><condition></td><td>16-bit mask</td><td>0 to 65535</td><td></td></tr>
<tr><td></td><td></td><td>Bit 0</td><td>Baseboard register test failed</td></tr>
<tr><td></td><td></td><td>Bit 1</td><td>Unused</td></tr>
<tr><td></td><td></td><td>Bit 2</td><td>Baseboard ROM test failed</td></tr>
<tr><td></td><td></td><td>Bit 3</td><td>Unused</td></tr>
<tr><td></td><td></td><td>Bit 4</td><td>Ref oscillator test failed</td></tr>
<tr><td></td><td></td><td>Bit 5</td><td>DRAM test failed</td></tr>
<tr><td></td><td></td><td>Bit 6</td><td>Flash memory test failed</td></tr>
<tr><td></td><td></td><td>Bit 7</td><td>Unused</td></tr>
<tr><td></td><td></td><td>Bit 8</td><td>AWG 1 test failed</td></tr>
<tr><td></td><td></td><td>Bit 9</td><td>AWG 2 test failed</td></tr>
<tr><td></td><td></td><td>Bits 10-15</td><td>Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Test Enable Command

Status Questionable Test Enable Query</td><td>
Sets or queries the contents of the Questionable Test Status Enable Register. The Questionable Test Status Enable Register enables the reporting of questionable test events to the Questionable Summary Register. The following considerations apply when using the Status Questionable Test Enable Command/Query:

The Questionable Test Status Enable Register is a bit mask that allows selected questionable self-test events to be reported to the Questionable Summary Register.
Only low to high (inactive to active) Questionable Test Status Register bit transitions are reported.
Questionable Test Status events report in bit 9 of the Questionable Summary Register.
The Status Preset Command sets the 15 LSB enable register bits to “1”, which enables all test event reporting.

Command Syntax
STATus:QUEStionable:TEST:ENABle <enable>

Query Syntax
STATus:QUEStionable:TEST:ENABle? → <enable>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th><th></th></tr>
<tr><td><enable></td><td>16-bit mask</td><td>0 to 65535</td><td></td></tr>
<tr><td></td><td></td><td>Bit 0</td><td>Baseboard register test failed</td></tr>
<tr><td></td><td></td><td>Bit 1</td><td>Unused</td></tr>
<tr><td></td><td></td><td>Bit 2</td><td>Baseboard ROM test failed</td></tr>
<tr><td></td><td></td><td>Bit 3</td><td>Unused</td></tr>
<tr><td></td><td></td><td>Bit 4</td><td>Ref oscillator test failed</td></tr>
<tr><td></td><td></td><td>Bit 5</td><td>DRAM test failed</td></tr>
<tr><td></td><td></td><td>Bit 6</td><td>Flash memory test failed</td></tr>
<tr><td></td><td></td><td>Bit 7</td><td>Unused</td></tr>
<tr><td></td><td></td><td>Bit 8</td><td>AWG 1 test failed</td></tr>
<tr><td></td><td></td><td>Bit 9</td><td>AWG 2 test failed</td></tr>
<tr><td></td><td></td><td>Bits 10-15</td><td>Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Test Event Query</td><td>
Queries the Questionable Test Status Event Register. The Questionable Test Status Event Register identifies unit tests that have completed with questionable results, such as self test errors. The following considerations apply when using the Status Questionable Test Event Query:

The Questionable Test Status Event Register records the history of the questionable test results generated since the previous Status Questionable Test Event Query.
The Status Questionable Test Event Query clears the Questionable Test Status Event Register after returning the current register contents.
Questionable Test Status Event Register reports in bit 9 of the Questionable Summary Register.
In order to identify questionable results from a particular process, the Questionable Test Status Event Register must be cleared by reading it before the processes are run.

Command Syntax
None

Query Syntax
STATus:QUEStionable:TEST[:EVENt]? → <event>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th><th></th></tr>
<tr><td><event></td><td>16-bit mask</td><td>0 to 65535</td><td></td></tr>
<tr><td></td><td></td><td>Bit 0</td><td>Baseboard register test failed</td></tr>
<tr><td></td><td></td><td>Bit 1</td><td>Unused</td></tr>
<tr><td></td><td></td><td>Bit 2</td><td>Baseboard ROM test failed</td></tr>
<tr><td></td><td></td><td>Bit 3</td><td>Unused</td></tr>
<tr><td></td><td></td><td>Bit 4</td><td>Ref oscillator test failed</td></tr>
<tr><td></td><td></td><td>Bit 5</td><td>DRAM test failed</td></tr>
<tr><td></td><td></td><td>Bit 6</td><td>Flash memory test failed</td></tr>
<tr><td></td><td></td><td>Bit 7</td><td>Unused</td></tr>
<tr><td></td><td></td><td>Bit 8</td><td>AWG 1 test failed</td></tr>
<tr><td></td><td></td><td>Bit 9</td><td>AWG 2 test failed</td></tr>
<tr><td></td><td></td><td>Bits 10-15</td><td>Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Test AWG Condition Query</td><td>
Queries the Questionable AWG Test Status Condition Register. The Questionable AWG Test Status Condition Register identifies current questionable results from all internally-generated AWG conditions. The following considerations apply when using the Status Questionable Test AWG Condition Query:

The Questionable AWG Test Status Condition Register identifies current AWG test conditions. Use the Status Questionable Test AWG Event Query to identify a history of which AWG conditions have failed since the last event status check.
The Status Questionable Test AWG Condition Query does not clear the Questionable AWG Test Status Condition Register.

Command Syntax
None

Query Syntax
STATus:QUEStionable:TEST:AWG<n>:CONDition? → <condition>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>1 AWG 1
2 AWG 2 (4 channel only)</td></tr>
<tr><td><condition></td><td>15-bit mask</td><td>0 to 65535
Bit 0 Register test failed
Bit 1 ROM test failed
Bit 2 Sample Clock test failed
Bit 3 Memory Clock test failed
Bits 4-5 Unused
Bit 6
AWG1 RAM1 test failed
AWG2 RAM3 test failed
Bit 7
AWG1 RAM2 test failed
AWG2 RAM4 test failed
Bit 8 Upload Bank A Failed
Bit 9 Upload Bank B Failed
Bit 10 Program DDS1 Failed
Bit 11 Program DDS2 Failed
Bit 12-14 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Test AWG Enable Command

Status Questionable Test AWG Enable Query</td><td>
Sets or queries the contents of the Questionable AWG Test Status Enable Register. The Questionable AWG Test Status Enable Register enables the reporting of questionable AWG test events to the Questionable Test Register. The following considerations apply when using the Status Questionable Test AWG Enable Command/Query:

The Questionable AWG Test Status Enable Register is a bit mask that allows selected questionable self-test events to be reported to the Questionable Test Register.
Only low to high (inactive to active) Questionable AWG Test Status Register bit transitions are reported.
Questionable AWG Test Status events report in bits 8 and 9 of the Questionable TEST Register.
The Status Preset Command sets the 15 LSB enable register bits to “1”, which enables all test event reporting.

Command Syntax
STATus:QUEStionable:TEST:AWG<n>:ENABle <enable>

Query Syntax
STATus:QUEStionable:TEST:AWG<n>:ENABle? → <enable>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 AWG 1
2 AWG 2 (4 channel only)</td></tr>
<tr><td><enable></td><td>8-bit mask</td><td>0 to 65535
Bit 0 Register test failed
Bit 1 ROM test failed
Bit 2 Sample Clock test failed
Bit 3 Memory Clock test failed
Bits 4-5 Unused
Bit 6
AWG1 RAM1 test failed
AWG2 RAM3 test failed
Bit 7
AWG1 RAM2 test failed
AWG2 RAM4 test failed
Bit 8 Upload Bank A Failed
Bit 9 Upload Bank B Failed
Bit 10 Program DDS1 Failed
Bit 11 Program DDS2 Failed
Bit 12-14 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Test AWG Event Query</td><td>
Queries the Questionable AWG Test Status Event Register. The Questionable AWG Test Status Event Register identifies unit tests that have completed with questionable results. The following considerations apply when using the Status Questionable Test AWG Event Query:

The Questionable AWG Test Status Event Register records the history of the questionable test results generated since the previous Status Questionable Test AWG Event Query.
The Status Questionable Test AWG Event Query clears the Questionable AWG Test Status Event Register after returning the current register contents.
Questionable AWG Test Status Event Register reports in bits 8 and 9 of the Questionable Test Register.
In order to identify questionable results from a particular process, the Questionable AWG Test Status Event Register must be cleared by reading it before the processes are run.

Command Syntax
None

Query Syntax
STATus:QUEStionable:TEST:AWG<n>[:EVENt]? → <event>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 AWG 1
2 AWG 2 (4 channel only)</td></tr>
<tr><td><event></td><td>8-bit mask</td><td>0 to 65535
Bit 0 Register test failed
Bit 1 ROM test failed
Bit 2 Sample Clock test failed
Bit 3 Memory Clock test failed
Bits 4-5 Unused
Bit 6
AWG1 RAM1 test failed
AWG2 RAM3 test failed
Bit 7
AWG1 RAM2 test failed
AWG2 RAM4 test failed
Bit 8 Upload Bank A Failed
Bit 9 Upload Bank B Failed
Bit 10 Program DDS1 Failed
Bit 11 Program DDS2 Failed
Bit 12-14 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Voltage Condition Query</td><td>
Queries the contents of the Questionable Voltage Status Condition Register. The Questionable Voltage Status Condition Register identifies the voltage overages for the output channels. The following considerations apply when using the Status Questionable Voltage Condition Query:

The Questionable Voltage Status Condition Register identifies voltage overloads. Use the Status Questionable Voltage Event Query to identify a history of which voltages have had overloads since the last voltage status check.
Questionable Voltage Status events report in bit 0 of the Questionable Summary Register.
The Status Questionable Voltage Condition Query does <u>not</u> clear the Questionable Voltage Status Condition Register.

Command Syntax
None

Query Syntax
STATus:QUEStionable:VOLTage:CONDition? → <condition>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><condition></td><td>16-bit mask</td><td>0 to 65535
Bits 0-3 Unused
Bit 4 AWG1 Pos Gain Undervoltage
Bit 5 AWG1 Neg Gain Undervoltage
Bit 6 AWG2 Pos Gain Undervoltage
Bit 7 AWG2 Neg Gain Undervoltage
Bit 8 Accessory 1 Fault
Bit 9 Accessory 2 Fault
Bit 10 Accessory 3 Fault
Bit 11 Accessory 4 Fault
Bits 12-15 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Voltage Enable Command

Status Questionable Voltage Enable Query</td><td>
Sets or queries the contents of the Questionable Voltage Status Enable Register. The following considerations apply when using the Status Questionable Voltage Enable Command/Query:

The Questionable Voltage Status Enable Register is a bit mask that allows selected questionable voltage events to be reported to the Questionable Summary Register.
Only low to high (inactive to active) Questionable Voltage Status Register bit transitions are reported.
Questionable Voltage Status events report in bit 0 of the Questionable Summary Register.
The Status Preset Command sets the 15 LSB enable register bits to “1”, which enables all voltage overload and over voltage reporting.

Command Syntax
STATus:QUEStionable:VOLTage:ENABle <enable>

Query Syntax
STATus:QUEStionable:VOLTage:ENABle? → <enable>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><enable></td><td>16-bit mask</td><td>0 to 65535
Bits 0-3 Unused
Bit 4 AWG1 Pos Gain Undervoltage
Bit 5 AWG1 Neg Gain Undervoltage
Bit 6 AWG2 Pos Gain Undervoltage
Bit 7 AWG2 Neg Gain Undervoltage
Bit 8 Accessory 1 Fault
Bit 9 Accessory 2 Fault
Bit 10 Accessory 3 Fault
Bit 11 Accessory 4 Fault
Bits 12-15 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Status Questionable Voltage Event Query</td><td>
Queries the Questionable Voltage Status Event Register. The Questionable Voltage Status Event Register identifies unit voltage overloads and over voltages that have completed with questionable results. The following considerations apply when using the Status Questionable Voltage Event Query:

The Questionable Voltage Status Event Register records the history of the questionable process results generated since the previous Status Questionable Voltage Event Query.
The Status Questionable Voltage Event Query clears the Questionable Voltage Status Event Register after returning the current register contents.
The Questionable Voltage Status Event Register reports in bit 0 of the Questionable Summary Register.
In order to identify questionable results from a particular process, the Questionable Voltage Status Event Register must be cleared by reading it before the processes are run.

Command Syntax
None
Query Syntax
STATus:QUEStionable:VOLTage [:EVENt]? → <event>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><event></td><td>16-bit mask</td><td>0 to 65535
Bits 0-3 Unused
Bit 4 AWG1 Pos Gain Undervoltage
Bit 5 AWG1 Neg Gain Undervoltage
Bit 6 AWG2 Pos Gain Undervoltage
Bit 7 AWG2 Neg Gain Undervoltage
Bit 8 Accessory 1 Fault
Bit 9 Accessory 2 Fault
Bit 10 Accessory 3 Fault
Bit 11 Accessory 4 Fault
Bits 12-15 Unused</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Sweep Direction Command

Sweep Direction Query</td><td>Sets or queries the channel's frequency sweep direction. Sweep Up will sweep from the Sweep Start Frequency to the Sweep Stop Frequency then repeat. Sweep Down will sweep from the Sweep Stop Frequency to the Sweep Start Frequency then repeat. Sweep Up and Down will sweep from the Sweep Start Frequency to the Sweep Stop Frequency and back to the Sweep Start Frequency.

Command Syntax
[SOURce<n>:]SWEep:DIRection <direction>

Query Syntax
[SOURce<n>:]SWEep:DIRection? → <direction>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><direction></td><td>Discrete</td><td>UP Up
DOWN Down
UDOWn Up and Down</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Sweep Frequency Start Command

Sweep Frequency Start Query</td><td>Sets or queries the channel's sweep frequencies. Sweep frequency range is limited to a 1000:1 start-to-stop ratio (3 decades). When in Function Mode these frequencies apply to the waveform cycle. When in Arbitrary or Sequence Mode these frequencies apply to the DAC Clock sample rate.

Command Syntax
[SOURce<n>:]SWEep:FREQuency <start>,<stop>

Query Syntax
[SOURce<n>:]SWEep:FREQuency? → <start>,<stop>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><start></td><td>Float</td><td>Function Mode 0.001Hz to 50 MHz
Arbitrary Mode 200S/s to 200MS/s
Sequence Mode 200S/s to 200MS/s</td></tr>
<tr><td><stop></td><td>Float</td><td>Function Mode 0.001Hz to 50 MHz
Arbitrary Mode 200S/s to 200MS/s
Sequence Mode 200S/s to 200MS/s</td></tr>
</table></td></tr>
<tr><td>Sweep Spacing Command

Sweep Spacing Query</td><td>Sets or queries the channel's sweep spacing type.

Command Syntax
[SOURce<n>:]SWEep:SPACing <type>

Query Syntax
[SOURce<n>:]SWEep:SPACing? → <type>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><type></td><td>Discrete</td><td>LINear Linear
LOGarithmic Logarithmic</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Sweep Time Command

Sweep Time Query</td><td>Sets or queries the sweep time. All frequency sweeps are completed in 1000 steps over the sweep time. Sweep time is common for all channels.

Command Syntax
[SOURce:]SWEep:TIME <time>

Query Syntax
[SOURce:]SWEep:TIME? → <time>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><time></td><td>Float</td><td>Time in seconds: 1 ms to 100 s</td></tr></table>
Resolution
1 µs</td></tr>
<tr><td>System Configure Query</td><td>Queries the current system configuration codes.

Command Syntax
None

Query Syntax
SYSTem:CONFigure? → <codes>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><codes></td><td>Array</td><td>Array of configuration codes</td></tr></table></td></tr>
<tr><td>System Error All Query</td><td>Returns all 32 entries in the error log and clears the error log. Multiple errors are stored sequentially in the error log with the oldest error first. A zero value is returned for all non-error entries when there are less than 32 errors stored in the error log.

Command Syntax
None

Query Syntax
SYSTem:ERRor:ALL? → <error_numbers>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><error_numbers></td><td>Integer Array</td><td>0 to –32768
0 No error
See Appendix 3, Error Codes, for a description of errors.</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>System Error Count Query</td><td>Returns the number of errors in the error log.

Command Syntax
None

Query Syntax
SYSTem:ERRor:COUNt? → <error_count>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><error_count></td><td>Integer</td><td>0 to 32</td></tr></table></td></tr>
<tr><td>System Error Next Query</td><td>Returns and clears the first entry in the error log. Multiple errors are stored sequentially in the error log with the oldest error first. A zero value is returned if there are no errors in the log.

Command Syntax
None

Query Syntax
SYSTem:ERRor[:NEXT]? → <error>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><error></td><td>Integer</td><td>0 to –32768
0 No error
See Appendix 3, Error Codes, for a description of errors.</td></tr></table></td></tr>
<tr><td>System Error Report Query</td><td>Returns the command string that caused the last error to occur.

Command Syntax
None

Query Syntax
SYSTem:ERRor:REPort? → <info>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><info></td><td>String</td><td>Up to 256 characters</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>System Identify Command

System Identify Query</td><td>Sets or queries the front panel LED identify state. When identify state is active, the instrument's identification LED will toggle.

Command Syntax
SYSTem:IDENtify <state>

Query Syntax
SYSTem:IDENtify?→<state>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td colspan="2">Range</td></tr>
<tr><td><state></td><td>Discrete</td><td>ON or 1
OFF or 0</td><td>Active state
Inactive state</td></tr>
</table>
</td></tr>
<tr><td>System Memory Query</td><td>Returns the amount of memory that is available for each of the output channels.

Command Syntax
None

Query Syntax
SYSTem:MEMory? → <size>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><size></td><td>Integer</td><td>Total Memory in Bytes</td></tr>
</table>
</td></tr>
<tr><td>System Memory Clear Command</td><td>Clears all non-volatile system memory.

Command Syntax
SYSTem:MEMory:CLEar

Query Syntax
None

Parameters
None</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>System Restore Command

System Restore Query</td><td>Selects or returns the configuration location used to initialize the instrument during boot up. Location 0 restores the instrument to default reset conditions. These locations can be saved to using the Save Instrument State Command.

Command Syntax
SYSTem:RESTore <number>

Query Syntax
SYSTem:RESTore? → <number>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><number></td><td>Integer</td><td>0 to 14</td></tr></table></td></tr>
<tr><td>System Temperature Query</td><td>Returns the current instrument temperature in degrees Celsius.

Command Syntax
None

Query Syntax
SYSTem:TEMPerature? → <degrees>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><degrees></td><td>Float</td><td>0 to 65 °C</td></tr></table></td></tr>
<tr><td>System Test Count Query</td><td>Returns the number of failure reports available from the last self-test.

Command Syntax
None

Query Syntax
SYSTem:TEST:COUNt? <location> → <count>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><location></td><td>Discrete</td><td>BASEboard Baseboard Self Test
SUBModule1 Channels 1-2
SUBModule2 Channels 3-4, ZT5xx2 only</td></tr><tr><td><count></td><td>Integer</td><td>Baseboard Count: 0-9
Submodule Count: 0-13</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>System Test Report Query</td><td>Returns an information string that can help identify what failed in an instrument self-test. When multiple failure reports are available they are returned in the order of occurrence. Test reports are not cleared when read, but are cleared when a new test is performed.

Command Syntax
None

Query Syntax
SYSTem:TEST:REPort? → <info>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><info></td><td>String</td><td>Up to 256 characters</td></tr></table></td></tr>
<tr><td>System Undo Command</td><td>Reverses instrument reset or state recall events.

Command Syntax
SYSTem:UNDO

Query Syntax
None

Parameters
None</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trace Copy Query</td><td>
Copies waveform data from the source channel to the destination channel. Trace copies preserve the data codes, but do not preserve any range information such as frequency or amplitude.
Trace Waveform Check Query must be called prior to copies that have WAVeform<n> as the destination.

Command Syntax
None

Query Syntax
TRACe:COPY? <source>,<dest> → <status>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><source></td><td>Discrete</td><td>OUTPut<n> Output channels, where <n> may be 1, 2, 3, or 4. Example: OUTP1.
REFerence<n> Reference channels, where <n> may be 1, 2, 3, or 4. Example: REF1.
WAVeform<n> Waveform Library location where <n> may be any valid handle from 1 - 4096. Example: WAV1</td></tr>
<tr><td><dest></td><td>Discrete</td><td>OUTPut<n> Output channels, where <n> may be 1, 2, 3, or 4. Example: OUTP1.
REFerence<n> Reference channels, where <n> may be 1, 2, 3, or 4. Example: REF1.
WAVeform<n> Waveform Library location where <n> may be any valid handle from 1 - 4096. Example: WAV1</td></tr>
<tr><td><status></td><td>Discrete</td><td>1 Copy successful
0 Copy failed</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trace Invert Query</td><td>Inverts waveform data in the source channel. Invert is a one-time command and the channel will not remain inverted if a new waveform is generated or uploaded to the channel.

Command Syntax
None

Query Syntax
TRACe:INVert? <source> → <status>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><source></td><td>Discrete</td><td>OUTPut<n> Output channels, where <n> may be 1, 2, 3, or 4. Example: OUTP1.
REFerence<n> Reference channels, where <n> may be 1, 2, 3, or 4. Example: REF1.
WAVeform<n> Waveform Library location where <n> may be any valid handle from 1 - 4096. Example: WAV1</td></tr>
<tr><td><status></td><td>Discrete</td><td>1 Inversion successful
0 Inversion failed</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trace Output Command

Trace Output Query</td><td>
Trace Output Command loads data from the block transfer buffer to the output channel memory buffer.
When uploading a waveform to the instrument, one or more block transfers (ex: zbind_blkout) should be used to fill the upload block transfer buffer with as many points as possible, and then the Trace Output Command should be used to load the data to the channel memory. For waveforms larger than the upload block transfer buffer size, refill the buffer and call the Trace Output Command with an offset. Trace Ready Query should be called between buffer fills to ensure that the instrument has had time to process the last buffer. The upload block transfer buffer can contain up to 64 Kibytes (65,536 bytes) of data in samples that are either 2 bytes (s16) or 4 bytes (f32) each.

Trace Output Query downloads waveform data from the output channel memory buffer to the block transfer buffer.
When downloading a waveform from the instrument, the Trace Output Query should be used to load the data from the channel memory to the download block transfer buffer, and then one or more block transfers (ex: zbind_blkin) should be used to download the data from the buffer. If there is too much data to fit in the buffer, use successive calls of the Trace Output Query with an offset value to load additional waveform sections to the block transfer buffer. The download block transfer buffer can contain up to 8 Mbytes (8,388,608 bytes) of data in samples that are either 2 bytes (s16) or 4 bytes (f32) each.

Command Syntax
TRACe: OUTPut<n> <total_size>,<offset>,<addr>,<buffer_points>

Query Syntax
TRACe: OUTPut<n>? <offset> → <total_size>,<addr>,<buffer_points>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><offset></td><td>Integer</td><td>Offset in Samples; the number of samples already uploaded/downloaded</td></tr>
<tr><td><total_size></td><td>Integer</td><td>Total number of samples in the waveform</td></tr>
<tr><td><addr></td><td>Address</td><td>Block transfer buffer waveform memory start address</td></tr>
<tr><td><buffer_points></td><td>Integer</td><td>The number of samples that are in the block transfer buffer</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trace Preamble Query</td><td>
Reads waveform information from the source channel.

Source: Selects the source to read the waveform preamble from/to.
Type: Returns the type of waveform: valid or invalid.
Points: Returns the number of points in the waveform.
Count: The Acquisition Count is always 1.
Time Interval: Returns the time interval between points.
Time Offset: Returns 0.
Voltage Interval: Returns the voltage resolution. When the data format is F32, this value is equal to the peak-to-peak voltage. When the data format is S16 this value is Vpp/2^{16}.
Voltage Offset: Returns the DC offset voltage for output channels, library and reference channels return 0.

Command Syntax
None

Query Syntax
TRACe:PREamble? <source> → <type>,<points>,<count>,<time_interval>,<time_offset>,<volt_interval>,<volt_offset>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><source></td><td>Discrete</td><td>OUTPut<n> Output channels, where <n> may be 1, 2, 3, or 4. Example: OUTP1.
REFerence<n> Reference channels, where <n> may be 1, 2, 3, or 4. Example: REF1.
WAVeform<n> Waveform Library location where <n> may be any valid handle from 1 – 4096. Example: WAV1</td></tr>
<tr><td><type></td><td>Discrete</td><td>0 Valid Waveform
1 Invalid Waveform</td></tr>
<tr><td><point></td><td>Integer</td><td>0 to maximum waveform size</td></tr>
<tr><td><count></td><td>Discrete</td><td>1</td></tr>
<tr><td><time_interval></td><td>Float</td><td>Time in seconds</td></tr>
<tr><td><time_offset></td><td>Float</td><td>0.0</td></tr>
<tr><td><volt_interval></td><td>Float</td><td>Voltage resolution in Volts</td></tr>
<tr><td><volt_offset></td><td>Float</td><td>Offset Voltage in Volts.</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trace Ready Query</td><td>This query is used to check the instrument ready state for long waveform uploads using the Trace Output Command, Trace Reference Command or Trace Waveform Command.

Command Syntax
None

Query Syntax
TRACe:READy? → <status>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><status></td><td>Discrete</td><td>1 Ready
0 Not Ready</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trace Reference Command

Trace Reference Query</td><td>
Trace Reference Command loads data from the block transfer buffer to the reference channel memory.
When uploading a waveform to the instrument, one or more block transfers (ex: zbind_blkout) should be used to fill the upload block transfer buffer with as many points as possible, and then the Trace Reference Command should be used to load the data to the reference channel memory. For waveforms larger than the upload block transfer buffer size, refill the buffer and call the Trace Reference Command with an offset. Trace Ready Query should be called between buffer fills to ensure that the instrument has had time to process the last buffer. The upload block transfer buffer can contain up to 64 Kibytes (65,536 bytes) of data in samples that are either 2 bytes (s16) or 4 bytes (f32) each.

Trace Reference Query downloads waveform data from the reference channel memory to the block transfer buffer.
When downloading a waveform from the instrument, the Trace Reference Query should be used to load the data from the reference channel memory to the download block transfer buffer, and then one or more block transfers (ex: zbind_blkin) should be used to download the data from the buffer. If there is too much data to fit in the buffer, use successive calls of the Trace Reference Query with an offset value to load additional waveform sections to the block transfer buffer. The download block transfer buffer can contain up to 8 Mibytes (8,388,608 bytes) of data in samples that are either 2 bytes (s16) or 4 bytes (f32) each.

Command Syntax
TRACe: REFerence<n> <total_size>,<offset>,<addr>,<buffer_points>

Query Syntax
TRACe: REFerence<n>? <offset> → <total_size>,<addr>,<buffer_points>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Reference Channel 1
2 Reference Channel 2
3 Reference Channel 3
4 Reference Channel 4</td></tr>
<tr><td><offset></td><td>Integer</td><td>Offset in Samples; the number of samples already uploaded/downloaded</td></tr>
<tr><td><total_size></td><td>Integer</td><td>Total number of samples in the waveform</td></tr>
<tr><td><addr></td><td>Address</td><td>Block transfer buffer waveform memory start address</td></tr>
<tr><td><buffer_points></td><td>Integer</td><td>The number of samples that are in the block transfer buffer</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trace Scale Query</td><td>
Scales the selected waveform data by the given factor. All scaling is relative to the current data, and not based ‘original’ data that existed prior to any manipulation such as inversion or scaling. Waveform data that is scaled up may become clipped and cannot be restored by scaling down.

Command Syntax
None

Query Syntax
TRACe:SCALe? <source>,<factor> → <status>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><source></td><td>Discrete</td><td>OUTPut<n> Output channels, where <n> may be 1, 2, 3, or 4. Example: OUTP1.
REFerence<n> Reference channels, where <n> may be 1, 2, 3, or 4. Example: REF1.
WAVeform<n> Waveform Library location where <n> may be any valid handle from 1 – 4096. Example: WAV1</td></tr>
<tr><td><factor></td><td>Float</td><td>Fraction of current waveform scale:
0.0 to 2.00
May also be passed as a percent:
0PCT to 200PCT</td></tr>
<tr><td><status></td><td>Discrete</td><td>1 Scale successful
0 Scale failed</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trace Waveform Command

Trace Waveform Query</td><td>
Trace Waveform Command loads data from the block transfer buffer to the waveform library. Trace Waveform Check Query must be called prior to loading a waveform.
When uploading a waveform to the instrument, one or more block transfers (ex: zbind_blkout) should be used to fill the upload block transfer buffer with as many points as possible, and then the Trace Waveform Command should be used to load the data to the library. For waveforms larger than the upload block transfer buffer size, refill the buffer and call the Trace Waveform Command with an offset. Trace Ready Query should be called between buffer fills to ensure that the instrument has had time to process the last buffer. The upload block transfer buffer can contain up to 64 Kibytes (65,536 bytes) of data in samples that are either 2 bytes (s16) or 4 bytes (f32) each.

Trace Waveform Query downloads waveform data from waveform library memory to the block transfer buffer.
When downloading a waveform from the instrument, the Trace Waveform Query should be used to load the data from the library to the download block transfer buffer, and then one or more block transfers (ex: zbind_blkin) should be used to download the data from the buffer. If there is too much data to fit in the buffer, use successive calls of the Trace Waveform Query with an offset value to load additional waveform sections to the block transfer buffer. The download block transfer buffer can contain up to 8 Mibytes (8,388,608 bytes) of data in samples that are either 2 bytes (s16) or 4 bytes (f32) each.

Command Syntax
TRACe:WAVeform<n> <total_size>,<offset>,<addr>,<buffer_points>

Query Syntax
TRACe:WAVeform<n>? <offset> → <total_size>,<addr>,<buffer_points>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Integer</td><td>Library handle: 1 to 4096</td></tr>
<tr><td><total_size></td><td>Integer</td><td>Total number of samples in the waveform</td></tr>
<tr><td><offset></td><td>Integer</td><td>Offset in Samples; the number of samples already uploaded/downloaded</td></tr>
<tr><td><addr></td><td>Address</td><td>Block transfer buffer waveform memory start address</td></tr>
<tr><td><buffer_points></td><td>Integer</td><td>The number of samples that are in the block transfer buffer</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trace Waveform Check Query</td><td>Returns a waveform library handle that has sufficient memory to fit a waveform of <points> samples. The return value will be zero if no library locations have sufficient memory. This command must be called before loading a waveform of size <points> into the library through either Trace Waveform Command or Trace Copy.

Command Syntax
None

Query Syntax
TRACe:WAVeform:CHECk? <points> → <handle>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><points></td><td>Integer</td><td>4 to 8 MiSamples</td></tr><tr><td><handle></td><td>Integer</td><td>Library handle: 1 to 4096</td></tr></table></td></tr>
<tr><td>Trace Waveform Clear Command</td><td>Clears library information associated with a specific handle. This also frees the waveform memory for use by other library waveforms.

Command Syntax
TRACe:WAVeform<n>:CLEar

Query Syntax
None

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><n></td><td>Integer</td><td>Library handle: 1 to 4096</td></tr></table></td></tr>
<tr><td>Trace Waveform Clear All Command</td><td>Clears all library information and frees all library memory.

Command Syntax
TRACe:WAVeform:CLEar:ALL

Query Syntax
None

Parameters
None</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trace Waveform Points Query</td><td>Returns the number of points in the specified waveform.

Command Syntax
None

Query Syntax
TRACe:WAVeform:POINts? <source> → <points>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><source></td><td>Discrete</td><td>OUTPut<n> Output channels, where <n> may be 1, 2, 3, or 4. Example: OUTP1.
REFerence<n> Reference channels, where <n> may be 1, 2, 3, or 4. Example: REF1.
WAVeform<n> Waveform Library location where <n> may be any valid handle from 1 – 4096. Example: WAV1</td></tr>
<tr><td><points></td><td>Integer</td><td>4 to Maximum Waveform Size</td></tr>
</table></td></tr>
<tr><td>Trace Waveform Valid Query</td><td>Queries whether the specified waveform library handle has valid waveform data associated with it.

Command Syntax
None

Query Syntax
TRACe:WAVeform<n>:VALid? → <valid>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Integer</td><td>Library handle: 1 to 4096</td></tr>
<tr><td><valid></td><td>Discrete</td><td>0 Waveform library handle unallocated
1 Waveform library handle allocated</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trigger Delay Command

Trigger Delay Query</td><td>Sets or queries the trigger delay. This time sets a delay between trigger detection and trigger event generation. This is used for Burst Mode only.

Command Syntax
TRIGger:DELay <time>

Query Syntax
TRIGger:DELay? → <time>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><time></td><td>Float</td><td>Time in seconds: 0.0 s to 6.5535 ms</td></tr></table>
Resolution
100 ns</td></tr>
<tr><td>Trigger External Impedance Command

Trigger External Impedance Query</td><td>Sets or queries the external trigger impedance. This value is used for Burst Trigger and Binary Modulation when source is set to External.

Command Syntax
TRIGger:EXTernal:IMPedance <imp>

Query Syntax
TRIGger:EXTernal:IMPedance? → <imp>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><imp></td><td>Discrete</td><td>50 or 1e6
MINimum 50 Ω
MAXimum 1 MΩ</td></tr></table></td></tr>
<tr><td>Trigger External Level Command

Trigger External Level Query</td><td>Sets or queries the external trigger level. This value is used for Burst Trigger and Binary Modulation when source is set to External.

Command Syntax
TRIGger:EXTernal:LEVel <level>

Query Syntax
TRIGger:EXTernal:LEVel? → <level>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><level></td><td>Discrete</td><td>Trigger level in volts: -2 V to 2 V
MINimum -2 V
MAXimum 2V</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trigger Internal Frequency Command

Trigger Internal Frequency Query</td><td>Sets or queries the internal trigger frequency. Channels in Burst Mode with Internal source during Initiate Continuous will output Burst Count waveform cycles at the frequency selected with this command.

Command Syntax
TRIGger:INTernal:FREQuency <freq>

Query Syntax
TRIGger:INTernal:FREQuency? → <freq>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><freq></td><td>Float</td><td>Frequency in Hertz: 0.01 Hz to 100 kHz
MINimum 0.01 Hz
MAXimum 100 kHz</td></tr></table>
Resolution
.01 Hz</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trigger Pattern Mask Command

Trigger Pattern Mask Query</td><td>Sets or queries which sources to use in the pattern. This value is used for Burst Trigger and Binary Modulation when source is set to Pattern.

Command Syntax
TRIGger:PATTern:MASK <pattern_mask>

Query Syntax
TRIGger:PATTern:MASK? → <pattern_mask>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><pattern_mask></td><td>16-bit mask</td><td>0 to 65535
0 Do not use in pattern trigger
1 Use in pattern trigger
Source Order (MSB–LSB):
• Bit 15 - Ignored
• Bit 14— ECLT1(VXI)
• Bit 13— ECLT0(VXI) / STAR(PCI)
• Bit 12—TTLT7
• Bit 11—TTLT6
• Bit 10—TTLT5
• Bit 9—TTLT4
• Bit 8—TTLT3
• Bit 7—TTLT2
• Bit 6—TTLT1
• Bit 5—TTLT0
• Bit 4—External Trigger
• Bits 3-0—Ignored</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trigger Pattern Truth Command

Trigger Pattern Truth Query</td><td>Sets or queries the state of each source necessary for the pattern trigger to occur. This value is used for Burst Trigger and Binary Modulation when source is set to Pattern.

Command Syntax
TRIGger:PATTern:TRUTh <pattern_truth>

Query Syntax
TRIGger:PATTern:TRUTh? → <pattern_truth>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><pattern_truth></td><td>16-bit mask</td><td>0 to 65535

0 Do not use in pattern trigger
1 Use in pattern trigger

Source Order (MSB–LSB):
• Bit 15 - Ignored
• Bit 14— ECLT1(VXI)
• Bit 13— ECLT0(VXI) / STAR(PCI)
• Bit 12—TTLT7
• Bit 11—TTLT6
• Bit 10—TTLT5
• Bit 9—TTLT4
• Bit 8—TTLT3
• Bit 7—TTLT2
• Bit 6—TTLT1
• Bit 5—TTLT0
• Bit 4—External Trigger
• Bits 3-0—Ignored</td></tr>
</table></td></tr>
<tr><td>Trigger Slope Command

Trigger Slope Query</td><td>Sets or queries the active trigger edge. This is only used in Burst Mode.

Command Syntax
TRIGger:SLOPe <slope>

Query Syntax
TRIGger:SLOPe? → <slope>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><slope></td><td>Discrete</td><td>POSitive Rising Edge
NEGative Falling Edge</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Trigger Source Command

Trigger Source Query</td><td>Sets or queries the trigger signal source. This is only used in Burst Mode.

Command Syntax
TRIGger:SOURce <source>

Query Syntax
TRIGger:SOURce? → <source>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><source></td><td>Discrete</td><td>EXTernal External trigger source
TTLTrg<n> TTL trigger line, where <n> may be 0, 1, 2, 3, 4, 5, 6, or 7
ECLTrg<n> VXIbus ECL trigger line, where <n> may be 0 or 1. VXI only
MANual Manual trigger
PATTern Pattern trigger
INTernal Internal trigger
STAR PCI only.</td></tr>
</table></td></tr>
<tr><td>Trigger Timestamp Query</td><td>Returns the trigger timestamp of the most recent trigger event in fractional seconds with a 1 second period.

Command Syntax
None

Query Syntax
TRIGger:TIMestamp? → <seconds>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><seconds></td><td>Float</td><td>0 to 1 second</td></tr>
</table>Resolution
100 ns</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Voltage Amplitude Command

Voltage Amplitude Query</td><td>
Sets or queries the channel's peak-to-peak voltage amplitude. The combination of Amplitude and Offset may not exceed ±14 V:

|Vamp/2 + Voffs| ≤ 14 V

When driving a low impedance (50Ω) load all output voltages are halved. Amplitudes less than 1 Vpp may be sent in Function Mode; the instrument will automatically scale the waveform codes. To get smaller voltages in Arbitrary or Sequence Mode use the Trace Scale Command.

Command Syntax
[SOURce<n>:]VOLTage[:LEVel][:IMMediate][:AMPLitude] <range>

Query Syntax
[SOURce<n>:]VOLTage[:LEVel][:IMMediate][:AMPLitude]? → <range>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><range></td><td>Float</td><td>Range in Volts peak-to-peak:
Arbitrary/Sequence Mode:
1 Vpp to 28 Vpp
Function Mode
15 mVpp to 28 Vpp</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Voltage Offset Command

Voltage Offset Query</td><td>Sets or queries the channel's output voltage offset. The combination of Amplitude and Offset may not exceed ±14 V:

|Vamp/2 + Voffs| ≤ 14 V

When driving a low impedance (50Ω) load all output voltages are halved.

Command Syntax
[SOURce<n>:]VOLTage[:LEVel][:IMMediate]:OFFSet <offset>

Query Syntax
[SOURce<n>:]VOLTage[:LEVel][:IMMediate]:OFFSet? → <offset>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><offset></td><td>Float</td><td>Offset in Volts: -14 V to 14 V</td></tr>
</table>
</td></tr>
<tr><td>Waveform Maximum Query</td><td>Queries the maximum number waveforms that can be stored in the waveform library.

Command Syntax
None

Query Syntax
[SOURce:]WAVeform:MAXimum? → <max>

Parameters
<table>
<tr><td>Name</td><td>Type</td><td>Range</td></tr>
<tr><td><max></td><td>Integer</td><td>4096</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Waveform Points Command

Waveform Points Query</td><td>Sets or queries the number of points in a channel's waveform. The command is typically used only for arbitrary waveforms. When using standard functions, this will override the default number of points.

Command Syntax
[SOURce<n>:]WAVeform:POINts <points>

Query Syntax
[SOURce<n>:]WAVeform:POINts? → <points>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><points></td><td>32-bit unsigned integer</td><td>4 to 32 MiSamples =
4 to 33,554,432 Samples</td></tr>
</table></td></tr>
<tr><td>Waveform Size Maximum Query</td><td>Queries the maximum size for a single waveform that can be stored in the waveform library.

Command Syntax
None

Query Syntax
[SOURce:]WAVeform:SIZE:MAXimum? → <max>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><max></td><td>Integer</td><td>32 M =
33,554,432</td></tr>
</table></td></tr>
<tr><td>Waveform Size Minimum Query</td><td>Queries the minimum size for a single waveform that can be stored in the waveform library.

Command Syntax
None

Query Syntax
[SOURce:]WAVeform:SIZE:MINimum? → <min>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><min></td><td>Integer</td><td>4</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Waveform Size Quantum Query</td><td>Queries the quantum size for arbitrary waveforms. Quantum is the minimum size step.

Command Syntax
None

Query Syntax
[SOURce:]WAVeform:SIZE:QUANtum? → <quantum>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><quantum></td><td>Integer</td><td>1</td></tr></table></td></tr>
<tr><td>Waveform Switch Mode Command

Waveform Switch Mode Query</td><td>Sets or queries the channel’s switch mode. When the switch mode is Seamless, the channel will begin playing new waveforms at the end of a current waveform cycle. When the switch mode is Instantaneous, the new waveform will begin playing as soon as it is available.

Command Syntax
[SOURce<n>:]WAVeform:SWITch:MODE <mode>

Query Syntax
[SOURce<n>:]WAVeform:SWITch:MODE? → <mode>

Parameters
<table><tr><td>Name</td><td>Type</td><td>Range</td></tr><tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr><tr><td><mode></td><td>Discrete</td><td>SEAMless Seamless
INSTantaneous Instantaneous</td></tr></table></td></tr>
</table>

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Width Command

Width Query</td><td>Sets or queries the channel’s positive pulse width. This parameter is used for square and pulse waveforms only.

Command Syntax
[SOURce<n>:]WIDTh <width>

Query Syntax
[SOURce<n>:]WIDTh? → <width>

Parameters
<table>
<tr><th>Name</th><th>Type</th><th>Range</th></tr>
<tr><td><n></td><td>Discrete</td><td>1 Output Channel 1
2 Output Channel 2
3 Output Channel 3
4 Output Channel 4</td></tr>
<tr><td><width></td><td>Float</td><td>Positive Width in seconds:
0 to period</td></tr>
</table>
Resolution
1 DAC Clock Cycle</td></tr>
</table>

Specifications

Outputs

Channels	2 (ZT5211) 4 (ZT5212)
Analog Bandwidth	DC to 10 MHz (±0.1dB passband flatness) DC to 25 MHz (–1dB bandwidth) DC to 50 MHz (–3dB bandwidth)
Lowpass Filters	50 MHz, 5-pole Bessel 10 MHz, 5-pole Bessel 1 MHz, 5-pole Bessel 100 kHz, 5-pole Bessel
Slew Rate [1]	> 2000 V/μs into 50Ω (50 MHz Filter)
Rise/Fall Time	< 7 ns for 10V step into 50Ω (50 MHz Filter)
Range Adjust [2]	15 mVpp to 28 Vpp into high impedance 7.5 mVpp to 14 Vpp into 50 Ω Range independently adjustable for each channel
DC Offset Adjust	0 to ±7V into 50Ω 0 to ±14V into high impedance
Output Voltage Limit [3]	|Vamp+Voffset| ≤ 7V into 50Ω |Vamp+Voffset| ≤ 14V into high impedance
Output Current Limit	±140 mA recommended operating maximum ±350 mA short circuit maximum
Range Resolution [4]	0.5 mVpp
Range Accuracy [4]	< ±(0.5% of Range + 10 mVpp) at 25 °C ambient
Range Drift [4]	< ±1.5 mVpp per °C
DC Offset Resolution [4]	0.5 mV
DC Offset Accuracy [4]	< ±(0.5% of Offset Setting + 2 mV) at 25 °C ambient
DC Offset Drift [4]	< ±75 μV per °C

[1] Harmonic distortion increases for Range-Bandwidth combination above 400 Vpp-MHz
[2] Full-scale range adjustment preserves function generator 14-bit DAC resolution and dynamic range
[3] Vamp = range/2, Voffset = DC offset
[4] Resolution, accuracy & drift specifications shown for high-impedance load, divide specifications by 2 for 50 Ω load

Output Impedance	50Ω typical
Connectors	BNC

Digital-to-Analog Converter (DAC)

DAC Resolution: 14 Bits (0.0061% of Full-Scale Range)

Range (Vpp) high impedance/no load	Range (Vpp) 50 Ω load	DAC resolution
28.0 – 1.0	14.0 – 0.5	14 bits
0.9995 – 0.5	0.4995 – 0.25	13 bits
0.4995 – 0.25	0.2495 – 0.125	12 bits
0.2495 – 0.125	0.1245 – 0.0625	11 bits
0.1245 – 0.0625	0.062 – 0.031	10 bits
0.062 – 0.031	0.0305 – 0.0155	9 bits
0.0305 – 0.015	0.015 – 0.0075	8 bits

Waveform Length	4 Sample to 32 MiSamples/channel
DAC Clock	DAC sample clocks generated by Direct Digital Synthesizers All channels have independent or common DAC sample clocks
DAC Clock Rates	200 Samples/s to 200 MSamples/s
DAC Clock Resolution	< 0.01 ppm or > 8 digits (i.e. 0.116 Hz from 20 MS/s to 200 MS/s)
DAC Clock Sweep	Swept DAC clock rate, linear or log sweep See Sweep Mode for additional details
DAC Clock Output	External Output: 100 Hz to 100 MHz common clock source
DAC Clock Jitter	< 20 ps
Timebase Reference	10 MHz
Timebase Reference Source	Internal TCXO, External Input, Backplane (PXI, VXI), Timing Expansion Connector (PCI)
Internal TCXO Timebase	± 2.5 ppm accuracy
Timebase Output	External Output, Timing Expansion Connector Reference I/O (PCI)
Channel-to-Channel Skew	< 500 ps difference between channels (50 MHz Filter)
Channel-to-Channel Isolation	≥ 60 dB

RMS Noise (with DAC clock above filter cutoff) ≤ 1mV into 50Ω (50 MHz Filter)

Spectral Purity (sine)

Output Level	Output Frequency	Harmonic	Non Harmonic
≤ +20 dBm	100 kHz	< -73 dBc	< -52 dBc
	20 MHz	< -50 dBc	< -52 dBc
	50 MHz	< -38 dBc	< -52 dBc
> +20 dBm	100 kHz	< -70 dBc	< -52 dBc
	20 MHz	< -30 dBc	< -52 dBc
	50 MHz	< -20 dBc	< -52 dBc

Operation Modes

Continuous Mode

Functionality: Generate output continuously when initiated

Burst Mode

Functionality: Generate a discrete number on cycles upon trigger event

Number of Cycles: 1 to 65535, programmable
A cycle is one waveform period or one waveform sequence

Binary Modulation Mode

Functionality: Toggle between two preloaded waveforms based upon modulation state,
Two unique waveforms preloaded into memory,
Enables Amplitude, Phase, Frequency Shift Keying, or Gated Output

Modulation Source: External Input, Bus Trigger 0-7, Star Trigger (PXI),
ECL Trigger 0-1 (VXI), Internal Trigger, Software

Sweep Mode

Functionality: Sweeps DAC clock rate for swept frequency of output signal
Programmable start frequency & stop frequency
Programmable up, down, or up & down modes

Sweep Types: Linear or logarithmic sweep

Sweep Range: 1000:1 maximum sweep frequency range (start-to-stop ratio)

Sweep Time: 1 ms to 100 s sweep time programmable, 1µs resolution

Trigger

Trigger Source	External Input, Bus Trigger 0-7, Star Trigger (PXI), ECL Trigger 0-1 (VXI) Pattern, Internal Trigger, Software
Edge Trigger Mode	Rising or falling edge
Pattern Trigger Mode	Pattern match true or false
Pattern Sources	External Input, Bus Trigger 0-7, Star Trigger (PXI), ECL Trigger 0-1 (VXI)
Trigger Latency	< (20 DAC clock periods + 100 ns)
Trigger Detection Jitter	±½ DAC clock period
Trigger Delay	Programmable delay after trigger event before start of waveform 0 to 6.5535 ms programmable, 100 ns resolution
Internal Trigger	Programmable internal trigger source, 10 µs to 100 s period, 100 ns resolution
Trigger Timestamp	Captures trigger event time, 1 second wrap period, 100 ns resolution

Arm

Functionality	Arm to qualify trigger event
Source	External Input, Bus Trigger 0-7, Star Trigger (PXI), ECL Trigger 0-1 (VXI), Software
Polarity	Positive or Negative

External Input

Functionality	External Trigger, External 10 MHz Timebase Reference, External Arm, or External Modulation Input
Maximum Input	±5 V (DC + peak AC), CAT I
Threshold Adjustment	±2V
Threshold Accuracy	±20 mV
Threshold Resolution	0.5 mV
Input Impedance	1 MΩ || 30 pF or 50 Ω
Impedance Accuracy	± 2%
Input Bandwidth	300 MHz typical 250 MHz minimum
Input Hysteresis	20 mV (overdrive required)

Connector	BNC (VXI, LXI) SMB (PCI, PXI)

Sync Outputs

Channels	2 (ZT5211) 4 (ZT5212)
Outputs	External Output, Bus Trigger 0-7, ECL Trigger 0-1 (VXI) Timing Expansion Connector I/O (PCI)
Time Resolution	5 ns to 500 µs (200 MHz to 2 kHz) Synchronized to DAC clock
Polarity	Programmable high or low pulses
Timing	Programmable location and width (in DAC clock samples)

External Output

Output Source	SYNC1-2, SYNC3-4 (ZT5212), Arm Event, Trigger Event, Generation Complete, Operation Complete, Master Status Event, Constant Level, Timebase Reference Clock, Common DAC Clock/2, Programmable Clock, Programmable Pulse
Output Level	TTL Compatible into High Impedance (≥ 200 Ω) ± 24 mA Output Drive Capability
Output Enable	Tri-State Output Capability
Programmable Clock	Clock Period: 26.667 ns to 100 seconds 50% Duty Cycle
Programmable Pulse	Pulse Repetition Interval: 26.667 ns to 100 seconds Pulse Width: 16.667 ns
DAC Clock	Half the common DAC sample clock: 100 Hz to 100 MHz
Programmable Width	Programmable active pulse width upon event for: Arm, Trigger, Generation Complete, Operation Complete, Master Status 20 ns to 163.83 ms, 10 ns resolution
Connector	BNC (VXI, LXI) SMB (PCI, PXI)

Event Outputs

Functionality	Event Output Signals
Outputs	Bus Trigger 0-7, ECL Trigger 0-1 (VXI) Timing Expansion Connector (PCI)

Source	SYNC1-2, SYNC3-4 (ZT5212), Arm Event, Trigger Event, Generation Complete, Operation Complete, Master Status Event, Constant Level
Programmable Width	Programmable active pulse width upon event for: Arm, Trigger, Generation Complete, Operation Complete, Master Status 20 ns to 163.83 ms, 10 ns resolution

Standard Functions

Sine

Frequency	0.001 Hz to 50 MHz
Initial Phase	0 to 360°

Square

Frequency	0.001 Hz to 20 MHz
Duty Cycle	0 to 100%
Initial Phase	0 to 360°

Triangle

Frequency	0.001 Hz to 20 MHz
Initial Phase	0 to 360°

Ramp (sawtooth)

Frequency	0.001 Hz to 20 MHz
Initial Phase	0 to 360°
Shape	Ramp Up or Down

DC

Amplitude	±100% of Maximum Range

Haversine

Frequency	0.001 Hz to 50 MHz
Initial Phase	0 to 360°

Havercosine

Frequency	0.001 Hz to 50 MHz
Initial Phase	0 to 360°

Half Cycle Sine

Frequency	0.001 Hz to 50 MHz
Initial Phase	0 to 360°

Pulse

Frequency	0.001 Hz to 20 MHz
Pulse Width	0 to (Period – 1 Data Clock cycle)
Rise/Fall Time	(1 Data Clock cycle) to (Period – 2 Data Clock cycles)
Initial Delay	0 to (Period – 2 Data Clock cycles)

Sinc Pulse

Frequency	0.001 Hz to 50 MHz
Sinc Frequency	Frequency to 50 MHz
Initial Phase	0 to 360°

Gaussian Pulse

Frequency	0.001 Hz to 20 MHz
Standard Deviation	(1 Data Clock cycle) to (Period/2)
Initial Phase	0 to 360°

Lorentz Pulse

Frequency	0.001 Hz to 50 MHz
Half Width	(1 Data Clock cycle) to (Period/2)
Initial Phase	0 to 360°

Noise

Period	1 µs to 1000 s
Noise Type	Uniform White

AM

Center Frequency	100 Hz to 50 MHz
Modulation Source	Internal
Modulation Frequency	1 Hz to smaller of 1 MHz or Center Frequency
Modulation Depth	0 to 100 percent
Modulation Shape	Sine, Square, Triangle, Ramp Up, Ramp Down

FM

Center Frequency	100 Hz to 50 MHz
Modulation Source	Internal
Modulation Frequency	1 Hz to smaller of 1 MHz or Center Frequency
Frequency Deviation	1 Hz to smaller of 1 MHz or Center Frequency
Modulation Shape	Sine, Square, Triangle, Ramp Up, Ramp Down

Multi-Tone

Frequencies	100 Hz to 50 MHz
Tone Resolution	100 Hz minimum
Number of Tones	1 to 16

Serial Data

Bit Rate Frequency	0.001 Hz to 20 MHz
Word Length	4 to 64-bits

Arbitrary Waveforms

Functionality	DAC Sample-by-Sample Arbitrary Waveform Synthesis

Waveform Sequences

Sequence	Predefines up to 8 sequences of arbitrary waveforms 2 to 4,096 waveform stages in sequence Each waveform repeated 1 to 65535 times within stage Waveforms stages from waveform library and reference channels Each waveform stage has unique waveform handle and loop number Amplitude, offset and DAC sample rate apply to entire sequence
Waveform Library	Predefines a set of arbitrary waveforms

	Up to 4,096 waveforms in waveform library Total arbitrary waveform library memory limited to 8 MiSamples (16 MiB)
Reference Channels	Predefines a set of non-volatile arbitrary waveforms Quantity 4 reference waveforms Each reference waveform limited to 32 KiSamples (64 KiB)

Waveform Operations

Upload	Waveform memory written by host Ping-pong buffers enable upload during active waveform generation
Download	Waveform memory read by host
Copy	Waveform memory copied from one location or type to another
Invert	Waveform DAC codes inverted (2s compliment)
Scale	Waveform DAC codes linearly adjusted by scale factor
Waveform Data Formats	16-bit signed integer, 32-bit floating point real number Intel or Motorola Byte Order

Instrument Setup Storage

Reset	Non-volatile storage of default instrument setup configuration
Undo	Returns to previous state prior to reset or recall command
Save & Recall	Non-volatile storage of 14 instrument setup configurations
Initial Power-On State	Configurable power-on condition to any valid saved instrument state State 0 initializes to factory default power-on condition

Status Reporting

IEEE-488.2 Device Status	Reporting Structure including Status Byte, Standard Event Registers, Questionable Registers, Operation Registers

PCI/PXI Data Interface

PCI Bus	33 MHz, 32 bit
PCI Data Transfer Rate	132 Mbyte/s burst, up to 120 Mbyte/s sustained [5]
PCI Voltage	Universal, +3.3V or +5V
PCI Compatibility	Version 2.2
PXI Compatiblity	PXI Standard Slot and PXI Express Hybrid Slot Compatible
PXI Signals (XJ4 connector)	PXI_TRIG0-7 input/output selectable PXI_STAR input 10 MHz reference input Left and right side buses not used
Primary ID	3712 ($0E80_{16}$)
Secondary ID	ZT5211: 5211 ($145B_{16}$) ZT5212: 5212 ($145C_{16}$)

VXI Data Interface

Command Interface	A16 message-based servant, SCPI compatible
Interrupt Operation	Programmable interrupter, Level 1–7
Data Interface	A16 register-based DMA D32 or D16 data transfer
Manufacturer ID	3712 ($0E80_{16}$)
Secondary ID	521 (209_{16})

LXI Data Interface

Command Interface	LAN 10/100, USB 2.0 Full-Speed 12 MB/s, SCPI compatible
Manufacturer ID	3712 ($0E80_{16}$)
Secondary ID	521 (209_{16})

[5] Sustained transfer rates are dependent upon host system configuration

PXI XJ4 Trigger & Clock Pin Usage

Pin A5	PXI Trigger 3	(TTL level bi-directional)
Pin A6	PXI Trigger 2	(TTL level bi-directional)
Pin A7	PXI Trigger 1	(TTL level bi-directional)
Pin B5	PXI Trigger 4	(TTL level bi-directional)
Pin B7	PXI Trigger 0	(TTL level bi-directional)
Pin C5	PXI Trigger 5	(TTL level bi-directional)
Pin D6	PXI Star Trigger	(TTL level input)
Pin E5	PXI Trigger 6	(TTL level bi-directional)
Pin E6	PXI CLK10	(TTL level input)
Pin E7	PXI Trigger 7	(TTL level bi-directional)

PCI Timing Expansion Connector Pin Usage

Pin 1	Reference	(TTL level bi-directional)
Pin 3	Star Trigger	(TTL level bi-directional)
Pin 5	Trigger 7	(TTL level bi-directional)
Pin 7	Trigger 6	(TTL level bi-directional)
Pin 9	Trigger 5	(TTL level bi-directional)
Pin 11	Trigger 4	(TTL level bi-directional)
Pin 13	Trigger 3	(TTL level bi-directional)
Pin 15	Trigger 2	(TTL level bi-directional)
Pin 17	Trigger 1	(TTL level bi-directional)
Pin 19	Trigger 0	(TTL level bi-directional)

VXIbus P2 Trigger & Clock Pin Usage

Pin A1	ECLTRG0	(ECL level bi-directional)
Pin A3	ECLTRG1	(ECL level bi-directional)
Pin A23	TTLTRG0*	(TTL level bi-directional)
Pin A24	TTLTRG2*	(TTL level bi-directional)
Pin A26	TTLTRG4*	(TTL level bi-directional)

Pin A27	TTLTRG6*	(TTL level bi-directional)
Pin C1	CLK10+	(ECL level input)
Pin C2	CLK10–	(ECL level input)
Pin C23	TTLTRG1*	(TTL level bi-directional)
Pin C24	TTLTRG3*	(TTL level bi-directional)
Pin C26	TTLTRG5*	(TTL level bi-directional)
Pin C27	TTLTRG7*	(TTL level bi-directional)

LED Indicators

READY	OFF: Hardware failure ON: Unit has passed power-up self-diagnostics TOGGLE: unit has an error pending in error queue
HOST or LAN	OFF: Interface fault ON: Normal interface operation TOGGLE: device identify command received
TRG	OFF: trigger event not detected ON/PULSE: trigger complete event detected
ACTIVE	OFF: Instrument Idle ON/PULSE: Data acquisition initiated
PWR (LXI only)	OFF: Instrument off ON: Instrument powered on
1588 (LXI only)	OFF: IEEE 1588 clock not synchronized or fault ON: clock locked as IEEE 1588 slave TOGGLE @ 1s: clock synchronized as IEEE 1588 master TOGGLE @ 2s: clock synchronized as IEEE 1588 grand master

Power

Power Supplies

Product Option	Platform	Voltage	Typical Current	Maximum Current
ZT5211	**PCI or PXI**	+3.3 VDC +5 VDC +12 VDC –12 VDC	3.28A 0.49A 0.21A 0.00A	4.72A 0.75A 0.67A 0.00A
	VXI	+5 VDC +12 VDC +24 VDC –2 VDC –5.2 VDC –12 VDC –24 VDC	3.10A 0.21A 0.00A 0.07A 0.26A 0.00A 0.00A	4.51A 0.67A 0.00A 0.08A 0.34A 0.00A 0.00A
	LXI	115 VAC	0.28A	0.33A
ZT5212	**VXI**	+5 VDC +12 VDC +24 VDC –2 VDC –5.2 VDC –12 VDC –24 VDC	4.55A 0.42A 0.00A 0.07A 0.37A 0.00A 0.00A	6.52A 1.34A 0.00A 0.08A 0.52A 0.00A 0.00A
	LXI	115 VAC	0.40A	0.49A

Total Cooling & Power Consumption

Product Option	Platform	Typical Cooling & Power	Maximum Cooling & Power
ZT5211	**PCI or PXI**	15.8W	27.5W
	VXI	17.9W	32.44W
	LXI	32W	38W
ZT5212	**VXI**	29.77W	51.42W
	LXI	45W	57W

Physical

PCI Physical size	Single-Slot Short PCI Card 7.65” x 0.85” x 4.97” (LxWxH) 19.43 cm x 2.16 cm x 12.62 cm (LxWxH)
PXI Physical size	Single-Wide 3U CompactPCI/PXI Instrument 8.25” x 0.79” x 5.25” (LxWxH) 22.23 cm x 2.01 cm x 13.34 cm (LxWxH)
VXI Physical size	Single-Wide C-size VXIbus Instrument 14.45” x 1.20” x 10.35” (LxWxH) 36.70 cm x 3.05 cm x 26.29 cm (LxWxH)
LXI Physical size	Half-Width 1U LXI Instrument 13.35” x 7.25” x 1.75” (LxWxH) 33.91 cm x 18.42 cm x 4.45 cm (LxWxH)
PCI Weight	1 lb. or 0.45 kg
PXI Weight	1 lb. or 0.45 kg
VXI Weight	3 lbs. or 1.4 kg
LXI Weight	4 lbs. or 1.8 kg

Temperature Range

Operating	0 °C to +50 °C Ambient (MIL-PRF28800F Class 3)
Storage	-40 °C to +75 °C Ambient (MIL-PRF28800F Class 3)
Over-Temperature	Automatic shutdown if internal temperature exceeds +65 °C
Calibration Range	+20 °C to +30 °C Ambient, after a 20 minute warm-up period, to meet all calibration specification accuracies.

Relative Humidity

Operating or Storage	5 to 95 ± 5%, non-condensing, up to +30 °C 5 to 75 ± 5%, non-condensing, +30 °C to +40 °C 5 to 45 ± 5%, non-condensing, above +40 °C

Altitude

Operating	Up to 5,000 m
Storage	Up to 15,000 m

Safety

This product is designed to meet the requirements of the following standard of safety for electrical equipment for measurement, control and laboratory use:

EN 61010-1

Electromagnetic Compatibility

CE Marking EN 61326-1:1997 with A1:1998 and A2:2001 Compliant

FCC Part 15 (Class A) Compliant

Emissions
EN 55011 Radiated Emissions, ISM Group 1, Class A, distance 10 m, emissions < 1 GHz
EN 55011 Conducted Emissions, Class A, emissions < 30 MHz Immunity
EN 61000-4-2 Electrostatic Discharge (ESD), 4 kV by Contact, 8 kV by Air
EN 61000-4-3 RF Radiated Susceptibility, 10 V/m
EN 61000-4-4 Electrical Fast Transient Burst (EFTB), 2 kV AC Power Lines
EN 61000-4-5 Surge
EN 61000-4-6 Conducted Immunity
EN 61000-4-8 Power Frequency Magnetic Field, 30 A/m
EN 61000-4-11 Voltage Dips and Interrupts

CE Compliance

This product meets the necessary requirements of applicable European Directives for CE Marking as follows:

73/23/EEC	Low Voltage Directive (Safety)
89/336/EEC	Electromagnetic Compatibility Directive (EMC)

See Declaration of Conformity for this product for additional regulatory compliance information.

LXI Conformance

This product's LXI models are conformant to the LXI Consortium's Functional Class C, Revision 1.2

Default Reset Conditions

Parameter	Default
Arm	Positive Polarity, Immediate Source
Binary Modulation	Manual Source, 0 Manual State, 0 View
Burst	Count: 1, Initiate Continuous ON
DAC Clock	200 MS/s Frequency, Independent Mode, Output Channel 1 Common Clock Source
Data Format	Normal Byte Order (MSB First), Signed 16-bit Integer Type
Data Floating Point Precision	2 digit Exponent, 5 digit Mantissa
ECL Trigger Outputs (VXI Only)	Disabled, Positive Polarity, Trigger Event Source
External Input	1 MΩ Impedance, 0.0V Level
External Output	Disabled, Positive Polarity, Pulse Source
External Output Pulse	Clock Mode, 10 ms Period
Function	Sine Shape, 1 MHz Frequency, 1 µs Period, 0 degrees Phase
Function AM	25 % Depth, 5 kHz Frequency, Sine Shape
Function Duty Cycle	50 % (Square or Pulse Functions Only)
Function FM	5 kHz Deviation, 5 kHz Frequency, Sine Shape
Function Gaussian	25 ns Standard Deviation
Function Lorentz	25 ns Half Width
Function Multi-Tone	Tone 1: ON/1 MHz, Tone 2: ON/2 MHz, Tones 3-16 OFF/3-16 MHz

Parameter	Default
Function Pulse	500 ns Width, 0 ns Leading Edge, 0 ns Trailing Edge
Function Serial Data	16-bit Word Length, 8421_{16} Data Word, 65.5 ns Bit Period
Function Sinc	10 MHz Sinc Frequency
Initiate	OFF
Mode	Continuous Operation, Function Output
Output Channel Enable	ON
Output Filter	50 MHz
Output Sync	Disabled, Positive Polarity, 0 % Start Position, 10 % Stop Position
Output Voltage	10.0 Volt Amplitude, 0.0 V Offset
Reference Oscillator Output (PCI, PXI)	OFF (Tri-Stated)
Reference Oscillator Source	Internal
Sequence	All Sequences Cleared
Status Enable	Service Request: All OFF (00_{16}) Standard Event: All OFF (00_{16}) Operation: All OFF (0000_{16}) Questionable: All OFF (0000_{16}) Questionable Frequency: All ON ($7FFF_{16}$) Questionable Test: All ON ($7FFF_{16}$) Questionable Test AWG1: All ON ($7FFF_{16}$) Questionable Test AWG2: All ON ($7FFF_{16}$) Questionable Voltage: All ON ($7FFF_{16}$)
Sweep	Up Direction, Linear Spacing, 100 kHz Start Frequency, 1 MHz Stop Frequency, 550 kHz Center Frequency 900 kHz Frequency Span 1 s Sweep Time
Trigger	Internal Source, Positive Polarity
Trigger Internal Frequency	10 Hz
Trigger Output Event Time	1 ms (for EXT, TTL, ECL Trigger Outputs)
Trigger Pattern	0000_{16} Mask, 0000_{16} Truth

Parameter	Default
TTL Trigger Outputs	Disabled, Positive Polarity, Trigger Event Source
Waveform Points	200 points
Waveform Switching Mode	Seamless (End of Cycle)
WTB Outputs (LXI Only)	Disabled, Positive Polarity, Trigger Event Source, Wired-or Mode

Error Codes

The instrument maintains an error queue containing codes for fault conditions encountered during unit operation. These codes are listed in the table below along with a brief description of the code definition. The error log may be read by using *System Error Next Query* or *System Error All Query.*

Code	Error Summary	Description
-100	Command error	A generic syntax error (only used when a more specific error does not apply)
-101	Invalid character	A syntactic element contains a character which is invalid for that type
-102	Syntax error	An unrecognized command or data type was encountered
-103	Invalid separator	The parser was expecting a separator and encountered an illegal character
-104	Data type error	The parser recognized a data element different than the one allowed
-105	Get not allowed	
-108	Parameter not allowed	More parameters were received than expected
-109	Missing parameter	Fewer parameters were received than expected
-110	Command header error	A generic error was detected in the command
-111	Header separator error	A character which was not a legal separator was encountered while parsing the command
-112	Mnemonic too long	The command contains too many characters
-113	Undefined header	The command is correct, but undefined for the specific instrument
-114	Header suffix out-of-range	The suffix number makes the command invalid
-118	Query not allowed	

Code	Error Summary	Description
-120	Numeric data error	A generic numeric syntax error (only used when a more specific error does not apply)
-121	Invalid char in number	An invalid character for the data type was encountered
-123	Exponent too large	The magnitude of the exponent was larger than 32000
-124	Too many digits	The mantissa of a decimal numeric data element contained more than 255 digits excluding zero
-128	Numeric data not allowed	A legal numeric data element was received, but the instrument does not accept one in this position in the command
-130	Suffix error	General command suffix error (only used when a more specific error does not apply)
-131	Invalid suffix	The command suffix is invalid for this instrument
-134	Suffix too long	The command suffix is too long
-138	Suffix not allowed	A suffix was encountered after a numeric element which does not allow suffixes
-140	Character data error	General character data element error (only used when a more specific error does not apply)
-141	Invalid character data	Either a invalid character in the parameter or the character data is not valid for this command
-144	Character data too long	The character parameter contains too many characters
-148	Character data not allowed	The character data is legal but not supported by this instrument
-150	String data error	General data string error (only used when a more specific error does not apply)
-151	Invalid string data	An invalid string command parameter
-158	String data not allowed	A string element was in the wrong place for this instrument command
-160	Block data error	General block data error (only used when a more specific error does not apply)

Code	Error Summary	Description
-161	Invalid block data	An invalid block data element was received for this instrument command
-168	Block data not allowed	Block data element not allowed by this instrument command at this parameter
-170	Expression error	General expression error (only used when a more specific error does not apply)
-171	Invalid expression	Invalid expression data element like unmatched parentheses or illegal character
-178	Expression data not allowed	A legal expression was encountered but is not allowed by this instrument in this command
-180	Macro error	General macro error (only used when a more specific error does not apply)
-181	Invalid outside macro	Indicates that a macro parameter placeholder was encountered outside a macro definition
-183	Invalid inside macro	Syntactically invalid message unit sequence
-184	Macro parameter error	The command inside the macro definition had the wrong number or type of parameters
-200	Execution error	General execution error (only used when a more specific error does not apply)
-201	Invalid while in local	Indicates that the command is not executable while the device is in local control
-202	Settings lost due to RTL	The settings were lost when the instrument was returned to local control
-203	Command protected	Indicates that a legal password protected program command or query could not be executed because the command was disabled
-204	Communication Timeout	Indicates that the driver communication timed out due to unresponsiveness or a lock out.
-210	Trigger Error	General trigger error
-211	Not ready for trigger	Indicates that a trigger was received by the instrument but was ignored because of timing considerations
-212	Not ready for arm	Indicates that a ARM was received by the instrument but was ignored

Code	Error Summary	Description
-213	Already initiated	Indicates that a measurement request was ignored because another measurement was already in progress
-214	Not ready for trigger	The trigger is deadlocked because a measurement result was requested before the instrument measurement was triggered
-220	Parameter error	General program parameter error (only used when a more specific error does not apply)
-221	Settings conflict	Indicates that a legal command was received by the instrument but could not be executed because of the current state of the instrument
-222	Data out of range	Indicates that a valid parameter was received but could not be executed because the parameter is out of range for the instrument
-223	Too much data	The command contained more data than the instrument memory could support
-224	Illegal parameter value	A value outside the list of possible values was received
-225	Out of memory	The instrument contains insufficient memory to perform the requested operation
-226	Lists not the same length	The lists do not have equal lengths
-230	Data corrupt or stale	New reading started but not completed resulting in invalid data
-231	Questionable data	Indicates that there is a problem with the instrument measurement accuracy
-232	Data has invalid format	The command tried to execute using an inappropriate data format or structure
-233	Incompatible version	Indicates that a file version or instrument version is not appropriate for this command
-240	Hardware error	Indicates that a general error occurred because there was a hardware problem in the instrument (only used when a more specific error does not apply)
-241	Hardware missing	Indicates that a command could not be executed because a hardware option is not present

Code	Error Summary	Description
-250	Mass storage error	General mass storage error (only used when a more specific error does not apply)
-251	Missing mass storage	The command could not be executed because an optional mass storage device was not present
-252	Missing media	The command could not be executed because of a missing media (disk) from a storage device
-253	Corrupt media	Indicates that the requested media is corrupt (bad or unformatted disk)
-254	Media full	Indicates that the requested media is full
-255	Directory full	Indicates that the requested media directory is full
-256	File name not found	Indicates that the command or query could not be executed because the requested file could not be found
-257	File name error	Indicates that the command or query could not be executed because the requested file was in error
-258	Media protected	Indicates that the requested media is protected
-260	Expression execution failed	General command expression error (only used when a more specific error does not apply)
-261	Math expression execution failed	Indicates that a command tried to perform an illegal math operation
-270	Macro execution error	General macro error (only used when a more specific error does not apply)
-271	Macro syntax error	The command could not be executed because there is an error within the syntax of the macro
-272	Macro execution error	The command could not be executed because there is an error within the macro definition
-273	Illegal macro label	The macro label is not valid for this instrument
-274	Macro parameter error	The macro definition improperly uses a macro parameter placeholder
-275	Macro definition too long	The string or block content of a macro was too long for the instrument

Code	Error Summary	Description
-276	Macro recursion error	The macro program data sequence could not be executed because the instrument found it to be recursive
-277	Macro redefinition not allowed	The command could not be executed because the macro label was already defined
-278	Macro header not found	Could not execute the macro because the macro was not previously defined
-280	Program error	General downloaded program error (only used when a more specific error does not apply)
-281	Can not create program	Indicates that an attempt to create a downloaded program was unsuccessful generally due to lack of memory
-282	Illegal program name	The command referenced a nonexistent program or attempted to redefine an existing program
-283	Illegal variable name	An attempt was made to reference a nonexistent program variable
-284	Program currently running	An attempt was made to redefine or delete an existing program while it was running
-285	Program syntax error	Indicates that a syntax error appears in a downloaded program
-286	Program runtime error	A runtime error exists in a downloaded program
-290	Memory usage error	Indicates that the user request has directly or indirectly caused an error related to memory
-291	Out of memory	The instrument memory is full
-292	Reference name does not exist	The reference name does not exist
-293	Reference name already exists	The reference name already exists
-294	Incompatible Type	Indicates that the type or structure of a memory item is inadequate
-300	Device specific error	General instrument error (only used when a more specific error does not apply)
-310	System error	Indicates that an instrument system error has occurred
-311	Memory error	Indicates a physical fault in the instruments memory, such as a parity fault

Code	Error Summary	Description
-312	PUD memory lost	Indicates that the protected user data in the instrument has been lost
-313	Calibration memory corrupted	Indicates that the instruments nonvolatile calibration memory has been lost or corrupted
-314	Configuration memory corrupted	Indicates that the instruments nonvolatile memory that was saved has been lost or corrupted
-315	Manufacturing info corrupted	Indicates that the instruments nonvolatile configuration memory has been lost or corrupted
-320	Storage Fault	Indicates that the firmware detected a fault when using data storage. Generally this error does not indicate a hardware error
-321	Out of memory for an internal operation	An internal operation needed more memory than was available
-330	Self test failed	The internal self test failed. This self test is either run on power up or by command
-340	Calibration failed	The instrument internal calibration failed
-350	Queue overflow	This code indicates that there is no room in the queue and an error occurred but was not recorded
-360	Communications error	General instrument communications error (only used when a more specific error does not apply)
-361	Parity error in program message	The serial port parity bit was not correct when data was received
-362	Framing error in program message	A serial port stop bit was not detected when data was received
-363	Input buffer overrun	The input buffer on a serial port overflowed with data caused by improper or nonexistent spacing
-400	Query error	General query error (only used when a more specific error does not apply)
-410	Query interrupt error	Indicates that a command was received before the query was fully executed
-420	Query un-terminated error	An incomplete query command was received

Code	Error Summary	Description
-430	Query deadlock error	The instrument is locked due to an incomplete query command
-440	Query un-terminated after indefinite response	Indicates that a query was received in the same command after a query requesting an indefinite response was executed
-500	Power on	The instrument has detected an off to on transition in its power system
-600	User request	The instrument has detected the activation of a user request for local control
-700	Request control	The instrument requested to become the active controller-in-charge
-800	Operation complete	The instrument has completed all selected pending operations
-1001	PLL unlocked	The instrument clock in not locked to the PLL
-1002	Boot Failed	Firmware boot failure detected
-1003	Wave Coerced	Waveform parameters coerced

Commands Index

The following are alphabetic lists of the commands for the instrument.

IEEE-488 Common Commands

Name	Command Syntax
Clear Status Command	*CLS
Event Status Enable Command/Query	*ESE
Event Status Register Query	*ESR?
Identification Query	*IDN?
Operation Complete Command/Query	*OPC
Recall Instrument State Command	*RCL
Reset Command	*RST
Save Instrument State Command	*SAV
Service Request Enable Command/Query	*SRE
Status Byte Query	*STB?
Trigger Immediate Command	*TRG
Test Query	*TST?
Wait to Continue Command	*WAI

SCPI Instrument Specific Commands

Name	Command Syntax
Abort Command	ABORt
Accessory Identify Query	[SOURce<n>:]ACCessory:IDENtify?
AM Depth Command/Query	[SOURce<n>:]AM[:DEPTh]
AM Frequency Command/Query	[SOURce<n>:]AM:FREQuency
AM Shape Command/Query	[SOURce<n>:]AM:SHAPe
Arm Command	ARM[:IMMediate]
Arm Query	ARM?
Arm Polarity Command/Query	ARM:POLarity

Name	Command Syntax
Arm Source Command/Query	ARM:SOURce
Binary Modulation View Command/Query	BMODulation:VIEW
Binary Modulation Source Command/Query	BMODulation:SOURce
Binary Modulation State Command/Query	BMODulation[:STATe]
Burst Count Command/Query	[SOURce<n>:]BURSt:COUNt
Calibration Date Query	CALibration:DATE?
Calibration Default Command	CALibration:DEFault
Calibration External Adjust Command	CALibration:EXTernal:ADJust
Calibration External Data Query	CALibration:EXTernal:DATA?
Calibration Gain Adjust Command	CALibration:GAIN<n>:ADJust
Calibration Gain Data Query	CALibration:GAIN<n>:DATA?
Calibration Gain Voltage Command/Query	CALibration:GAIN<n>:VOLTage
Calibration Offset Adjust Command	CALibration:OFFSet<n>:ADJust
Calibration Offset Data Query	CALibration:OFFSet<n>:DATA?
Calibration Offset Voltage Command/Query	CALibration:OFFSet<n>:VOLTage
Calibration Restore Command	CALibration:RESTore
Calibration Reference Oscillator Adjust Command	CALibration:ROSCillator:ADJust
Calibration Reference Oscillator Data Query	CALibration:ROSCillator:DATA?
Calibration Save Command	CALibration:SAVE
DAC Clock Common Command/Query	[SOURce:]DAC:CLOCk:COMMon
DAC Clock Frequency Command/Query	[SOURce<n>:]DAC:CLOCk:FREQuency
DAC Clock Mode Command/Query	[SOURce<n>:]DAC:CLOCk:MODE
Duty Cycle Command/Query	[SOURce<n>:]DCYCle
Filter Frequency Command/Query	[SOURce<n>:]FILTer[:LPASs]:FREQuency
FM Deviation Command/Query	[SOURce<n>:]FM[:DEViation]
FM Frequency Command/Query	[SOURce<n>:]FM:FREQuency
FM Shape Command/Query	[SOURce<n>:]FM:SHAPe
Format Bye Order Command/Query	FORMat:BORDer
Format Data Command/Query	FORMat:DATA
Format Precision Command/Query	FORMat:PRECision
Frequency Command/Query	[SOURce<n>:]FREQuency[:CW]
Function Shape Command/Query	[SOURce<n>:]FUNCtion[:SHAPe]
Gaussian Standard Deviation Command/Query	[SOURce<n.:]GAUSsian:SDEViation
Initiate Continuous Command/Query	INITiate:CONTinuous
Initiate Command	INITiate[:IMMediate]

Name	Command Syntax
Initiate Query	INITiate?
Lorentz Half Width Command/Query	[SOURce<n>:]LORentz:HWIDth
Multi-Tone Clear Command	[SOURce<n>:]MTONe:CLEar
Multi-Tone Tone Frequency Command/Query	[SOURce<n>:]MTONe:TONE<n>:FREQuency
Multi-Tone Tone State Command/Query	[SOURce<n>:]MTONe:TONE<n>[:STATe]
Operation Mode Command/Query	[SOURce<n>:]OPERation:MODE
Output Mode Command/Query	[SOURce<n>:]OUTPut:MODE
Output State Command/Query	OUTPut<n>[:STATe]
Output Source Query	OUTPut<n>:SOURce?
Output ECL Trigger Polarity Command/Query (VXI only)	OUTPut:ECLTrg<n>:POLarity
Output ECL Trigger Source Command/Query (VXI only)	OUTPut:ECLTrg<n>:SOURce
Output ECL Trigger State Command/Query (VXI only)	OUTPut:ECLTrg<n>[:STATe]
Output Event Time Command/Query	OUTPut:EVENt[:TIME]
Output External Polarity Command/Query	OUTPut:EXTernal:POLarity
Output External Pulse Mode Command/Query	OUTPut:EXTernal:PULSe:MODE
Output External Pulse Period Command/Query	OUTPut:EXTernal:PULSe:PERiod
Output External Source Command/Query	OUTPut:EXTernal:SOURce
Output External State Command/Query	OUTPut:EXTernal[:STATe]
Output LXI Mode Command/Query (LXI only)	OUTPut:LXI<n>:MODE
Output LXI Polarity Command/Query (LXI only)	OUTPut:LXI<n>:POLarity
Output LXI Source Command/Query (LXI only)	OUTPut:LXI<n>:SOURce
Output LXI State Command/Query (LXI only)	OUTPut:LXI<n>[:STATe]
Output Sync On Position Command/Query	OUTPut:SYNC<n>:ON:POSition
Output Sync Off Position Command/Query	OUTPut:SYNC<n>:OFF:POSition
Output TTL Trigger Polarity Command/Query	OUTPut:TTLTrg<n>:POLarity
Output TTL Trigger Source Command/Query	OUTPut:TTLTrg<n>:SOURce
Output TTL Trigger State Command/Query	OUTPut:TTLTrg<n>[:STATe]
Period Command/Query	[SOURce<n>:]PERiod
Phase Command/Query	[SOURce<n>:]PHASe[:ADJust]
Pulse Transition Leading Command/Query	[SOURce<n>:]PULSe:TRANsition[:LEADing]
Pulse Transition Trailing Command/Query	[SOURce<n>:]PULSe:TRANsition:TRAiling
Reference Oscillator Frequency Query	[SOURce:]ROSCillator:FREQuency?

Name	Command Syntax
Reference Oscillator Output State Command/Query (PCI only)	[SOURce:]ROSCillator:OUTPut[:STATe]
Reference Oscillator Source Command/Query	[SOURce:]ROSCillator:SOURce
Sense ECL Trigger State Query (VXI only)	SENSe:ECLTrg<n>[:STATe]?
Sense LXI State Query (LXI only)	SENSe:LXI<n>[:STATe]?
Sense TTL Trigger State Query	SENSe:TTLTrg<n>[:STATe]?
Sequence Clear Command	[SOURce:]SEQuence:CLEar
Sequence Clear All Command	[SOURce:]SEQuence:CLEar:ALL
Sequence Data Loop Count Command/Query	[SOURce:]SEQuence:DATA:LCOunt
Sequence Data Waveform Command/Query	[SOURce:]SEQuence:DATA:WAVeform
Sequence Generate Query	[SOURce<n>:]SEQuence:GENerate?
Sequence Loop Maximum Query	[SOURce:]SEQuence:LOOP:MAXimum?
Sequence Maximum Query	[SOURce:]SEQuence:MAXimum?
Sequence Size Command/Query	[SOURce:]SEQuence:SIZE
Sequence Size Maximum Query	[SOURce:]SEQuence:SIZE:MAXimum?
Sequence Size Minimum Query	[SOURce:]SEQuence:SIZE:MINimum?
Serial Data Bit Period Command/Query	[SOURce<n>:]SDATa:BPERiod
Serial Data Word Length Command/Query	[SOURce<n>:]SDATa:WLENgth
Serial Data Word Command/Query	[SOURce<n>:]SDATa:WORD
Sinc Frequency Command/Query	[SOURce<n>:]SINC:FREQuency
Status Interrupt Request State Command/Query	STATus:IRQ[:STATe]
Status Operation Condition Query	STATus:OPERation:CONDition?
Status Operation Enable Command/Query	STATus:OPERation:ENABle
Status Operation Event Query	STATus:OPERation[:EVENt]?
Status Preset Command	STATus:PRESet
Status Questionable Calibration Condition Query	STATus:QUEStionable:CALibration:CONDition?
Status Questionable Calibration Enable Command/Query	STATus:QUEStionable:CALibration:ENABle
Status Questionable Calibration Event Query	STATus:QUEStionable:CALibration[:EVENt]?
Status Questionable Condition Query	STATus:QUEStionable:CONDition?
Status Questionable Enable Command/Query	STATus:QUEStionable:ENABle
Status Questionable Event Query	STATus:QUEStionable[:EVENt]?
Status Questionable Frequency Condition Query	STATus:QUEStionable:FREQuency:CONDition?
Status Questionable Frequency Enable Command/Query	STATus:QUEStionable:FREQuency:ENABle
Status Questionable Frequency Event Query	STATus:QUEStionable:FREQuency[:EVENt]?

Name	Command Syntax
Status Questionable Test Condition Query	STATus:QUEStionable:TEST:CONDition?
Status Questionable Test Enable Command/Query	STATus:QUEStionable:TEST:ENABle
Status Questionable Test Event Query	STATus:QUEStionable:TEST[:EVENt]?
Status Questionable Test AWG Condition Query	STATus:QUEStionable:TEST:AWG<n>:CONDitio n?
Status Questionable Test AWG Enable Command/Query	STATus:QUEStionable:TEST:AWG<n>:ENABle
Status Questionable Test AWG Event Query	STATus:QUEStionable:TEST:AWG<n>[:EVENt]?
Status Questionable Voltage Condition Query	STATus:QUEStionable:VOLTage:CONDition?
Status Questionable Voltage Enable Command/Query	STATus:QUEStionable:VOLTage:ENABle
Status Questionable Voltage Event Query	STATus:QUEStionable:VOLTage[:EVENt]?
Sweep Direction Command/Query	[SOURce<n>:]SWEep:DIRection
Sweep Frequency Command/Query	[SOURce<n>:]SWEep:FREQuency
Sweep Spacing Command/Query	[SOURce<n>:]SWEep:SPACing
Sweep Time Command/Query	[SOURce:]SWEep:TIME
System Configure Query	SYSTem:CONFigure?
System Error All Query	SYSTem:ERRor:ALL?
System Error Count Query	SYSTem:ERRor:COUNt?
System Error Next Query	SYSTem:ERRor[:NEXT]?
System Error Report Query	SYSTem:ERRor:REPort?
System Identify Command/Query	SYSTem:IDENtify
System Memory Query	SYSTem:MEMory?
System Memory Clear Command	SYSTem:MEMory:CLEar
System Restore Command/Query	SYSTem:RESTore
System Temperature Query	SYSTem:TEMPerature?
System Test Report Query	SYSTem:TEST:REPort?
System Undo Command	SYSTem:UNDO
Trace Copy Query	TRACe:COPY?
Trace Invert Query	TRACe:INVert?
Trace Output Command/Query	TRACe:OUTPut<n>
Trace Preamble Query	TRACe:PREamble?
Trace Ready Query	TRACE:READy?
Trace Reference Command/Query	TRACe:REFerence<n>
Trace Scale Query	TRACe:SCALe?

Name	Command Syntax
Trace Waveform Command Query	TRACe:WAVeform<n>
Trace Waveform Check Query	TRACe:WAVeform:CHECk?
Trace Waveform Clear Command	TRACe:WAVeform<n>:CLEar
Trace Waveform Clear All Command	TRACe:WAVeform:CLEar:ALL
Trace Waveform Points Query	TRACe:WAVeform:POINts?
Trace Waveform Valid Query	TRACe:WAVeform<n>:VALid?
Trigger Delay Command/Query	TRIGger:DELay
Trigger External Impedance Command/Query	TRIGger:EXTernal:IMPedance
Trigger External Level Command/Query	TRIGger:EXTernal:LEVel
Trigger Internal Frequency Command/Query	TRIGger:INTernal:FREQuency
Trigger Pattern Mask Command/Query	TRIGger:PATTern:MASK
Trigger Pattern Truth Command/Query	TRIGger:PATTern:TRUTh
Trigger Slope Command/Query	TRIGger:SLOPe
Trigger Source Command/Query	TRIGger:SOURce
Trigger Timestamp Query	TRIGger:TIMestamp
Voltage Amplitude Command/Query	[SOURce<n>:]VOLTage[:LEVel][:IMMediate][:AMPLitude]
Voltage Offset Command/Query	[SOURce<n>:]VOLTage[:LEVel][:IMMediate]:OFFSet
Waveform Maximum Query	[SOURce:]WAVeform:MAXimum?
Waveform Points Command/Query	[SOURce<n>:]WAVeform:POINts
Waveform Size Maximum Query	[SOURce:]WAVeform:SIZE:MAXimum?
Waveform Size Minimum Query	[SOURce:]WAVeform:SIZE:MINimum?
Waveform Size Quantum Query	[SOURce:]WAVeform:SIZE:QUANtum?
Waveform Switch Mode Command/Query	[SOURce<n>:]WAVeform:SWITch:MODE
Width Command/Query	[SOURce<n>:]WIDTh

ZTEC® Instruments